www.zte.com.cn/magazine/English

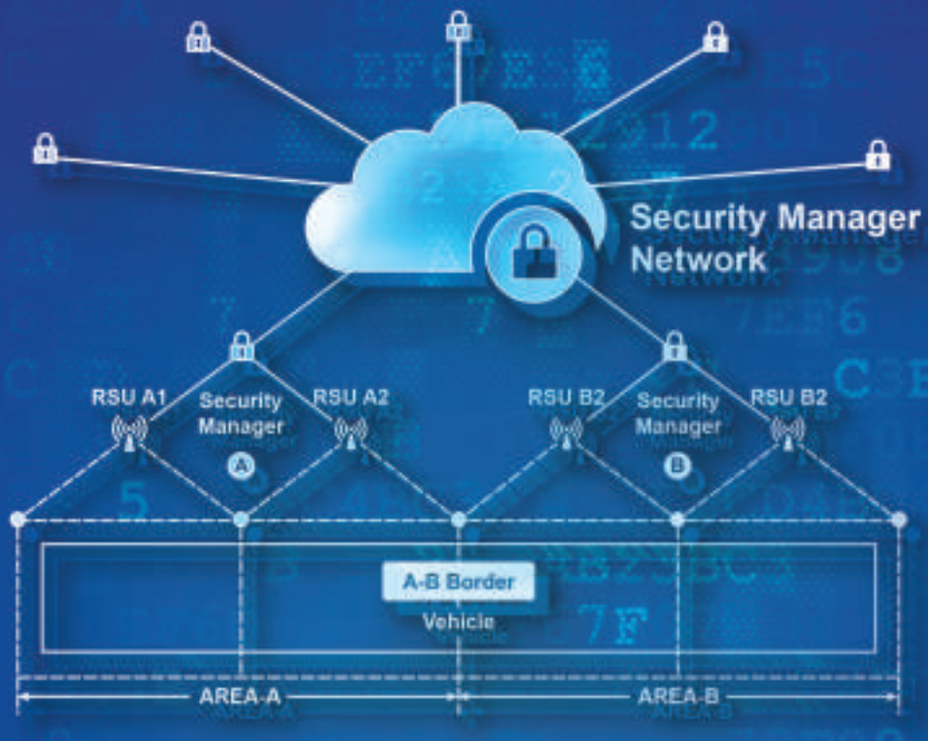
ISSN 1673-5188
CN 34-1294/ TN
CODEN ZCTOAK

ZTE COMMUNICATIONS

An International ICT R&D Journal Sponsored by ZTE Corporation

June 2016, Vol. 14 No. S0

SPECIAL TOPIC:
Recent Development on Security and Privacy in Modern Communication Environments

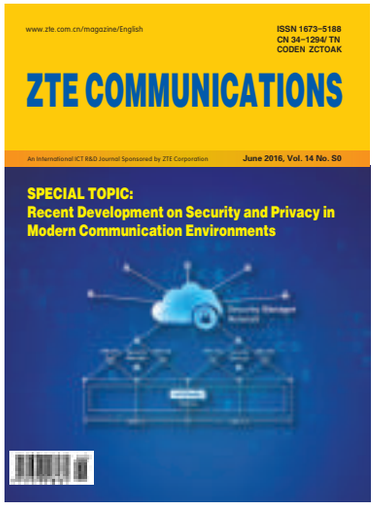


9 771673 518161 06

ZTE中兴 ZTE COMMUNICATIONS
VOLUME 14 NUMBER S0 JUNE 2016

ZTE Communications Editorial Board

CONTENTS



Special Topic: Recent Development on Security and Privacy in Modern Communication Environments

CONTENTS

ZTE COMMUNICATIONS
Vol. 14 No. S0 (Issue 51)
Quarterly
First English Issue Published in 2003

Supervised by:
Anhui Science and Technology Department

Sponsored by:
Anhui Science and Technology Information Research Institute and ZTE Corporation

Staff Members:
Editor-in-Chief: CHEN Jie
Executive Associate Editor-in-Chief: HUANG Xinming
Editor-in-Charge: ZHU Li
Editors: XU Ye, LU Dan, ZHAO Lu
Producer: YU Gang
Circulation Executive: WANG Pingping
Assistant: WANG Kun

Editorial Correspondence:
Add: 12F Kaixuan Building,
329 Jinzhai Road,
Hefei 230061, P. R. China
Tel: +86-551-65533356
Fax: +86-551-65850139
Email: magazine@zte.com.cn

Published and Circulated (Home and Abroad) by:
Editorial Office of
ZTE Communications

Printed by:
Hefei Tiancai Color Printing Company

Publication Date:
June 25, 2016

Publication Licenses:

ISSN 1673-5188
CN 34-1294/ TN

Advertising License:
皖合工商广字0058号

Annual Subscription:
RMB 80

Guest Editorial

ZHOU Wanlei and MIN Geyong

Recent Development on Security and Privacy in Modern Communication Environments

▶ ZHOU Wanlei

ZHOU Wanlei received the BEng and MEng degrees from Harbin Institute of Technology, China in 1982 and 1984 and the PhD degree from The Australian National University, Australia in 1991, all in computer science and engineering. He also received a DSc degree (a higher Doctorate degree) from Deakin University, Australia in 2002. He is currently the Alfred Deakin Professor (the highest honour the University can bestow on a member of academic staff) and Chair Professor in Information Technology, School of Information Technology, Deakin University. Professor Zhou was the head of School of Information Technology twice and associate dean of Faculty of Science and Technology in Deakin University. Before joining Deakin University, Professor Zhou served as a lecturer in University of Electronic Science and Technology of China, a system programmer in HP at Massachusetts, USA, a lecturer in Monash University, Australia, and a lecturer in National University of Singapore, Singapore. His research interests include distributed systems, network security, bioinformatics, and e- learning. Professor Zhou has published more than 300 papers in refereed international journals and refereed international conferences proceedings. He has also chaired many international conferences. Prof Zhou is a senior member of the IEEE.

▶ MIN Geyong

MIN Geyong is a professor of High Performance Computing and Networking in the Department of Mathematics and Computer Science, the College of Engineering, Mathematics and Physical Sciences at the University of Exeter, UK. He received the PhD degree in computing science from the University of Glasgow, UK in 2003, and the BSc degree in computer science from Huazhong University of Science and Technology, China in 1995. His research interests include future internet, computer networks, wireless communications, multimedia systems, information security, high performance computing, ubiquitous computing, modelling and performance engineering.

Nowadays, many emerging technologies have constructed modern communication systems, such as the internet, wired/wireless networks, sensor networks, RFID systems, cloud services and machine-to-machine interfaces. Modern communication allows billions of objects in the physical world as well as virtual environments to exchange data with each other in an autonomous way so as to create smart environments for transportation, healthcare, logistics, environmental monitoring, and many others. However, modern communication also introduces new challenges for the security of systems and processes and the privacy of individuals. Protecting information in modern communication environments is a complex and difficult task. Modern communication environments usually offer global connectivity and accessibility, which means anytime and anyway access and results in that the number of attack vectors available to malicious attackers might become incredibly large. Moreover, the inherent complexity of modern communication environments, where multiple heterogeneous entities located in different contexts can exchange information with each other, further complicates the design and deployment of efficient, interoperable, and scalable security mechanisms. The ubiquitous and cloud computing also makes the problem of privacy leakage serious. As a result, there is an increasing demand for developing new security and privacy approaches to guarantee the security, privacy, integrity, and availability of resources in modern communication environments.

This special issue includes six articles and can be categorised in 3 themes:

The first theme is review. The paper by YANG Mengmeng, ZHU Tianqing, ZHOU Wanlei, and XIANG Yang presents a literature survey on the attack models and countermeasures for privacy-preserving in social networks.

The second theme is secure applications. The paper by Faizal Riaz-ud-Din and Robin Doss presents a verification scheme for existential substring searches on text files stored on untrusted clouds to satisfy the desired properties of authenticity, completeness, and freshness. The paper by LEI Ao, Chibueze Ogah, Philip Asuquo, Haitham Cruickshank, and SUN Zhili presents a framework for providing secure key management within heterogeneous vehicular communication systems. Besides, the paper by LI Shancang and Imed Romdhani shows how to substantially improve the strength of passwords based on the analysis of text-password entropy.

The third theme is security and privacy for Android devices. The paper by CHEN Kuan-Lin and YANG Chung-Huang presents a privacy impact assessment framework for Android mobile devices to manage user privacy risks. The paper by DONG Zhenjiang, WANG Wei, LI Hui, et al. presents a security enhancement system with online authentication for android APK to improve the security level of the APK and it ensures a good balance between security and usability.

We hope this special issue will benefit the research and development community towards identifying challenges and disseminating the latest methodologies and solutions to security and privacy issues in modern communication environments. We sincerely thank all the authors who have submitted their valuable manuscripts to this special issue and all the reviewers who spent their precious time going through and commenting on the submitted manuscripts.

DOI: 10.3969/j. issn. 1673-5188. 2016. S0. 001
http://www.cnki.net/kcms/detail/34.1294.TN.20160615.1654.002.html, published online June 15, 2016

Attacks and Countermeasures in Social Network Data Publishing

YANG Mengmeng, ZHU Tianqing, ZHOU Wanlei, and XIANG Yang
(School of Information Technology, Deakin University, Burwood, VIC 3125, Australia)

Abstract

With the increasing prevalence of social networks, more and more social network data are published for many applications, such as social network analysis and data mining. However, this brings privacy problems. For example, adversaries can get sensitive information of some individuals easily with little background knowledge. How to publish social network data for analysis purpose while preserving the privacy of individuals has raised many concerns. Many algorithms have been proposed to address this issue. In this paper, we discuss this privacy problem from two aspects: attack models and countermeasures. We analyse privacy concerns, model the background knowledge that adversary may utilize and review the recently developed attack models. We then survey the state-of-the-art privacy preserving methods in two categories: anonymization methods and differential privacy methods. We also provide research directions in this area.

Keywords

social network; data publishing; attack model; privacy preserving

1 Introduction

Social network is a very popular platform where people make new friends and share their interests. A dramatically increasing number of users have joined social networks. Social network service providers hold a large amount of data. To some extent, these data provide a great opportunity to analyse social networks, while at the same time, it brings privacy concern.

Normally, in order to preserving users' privacy, social network data are published without identity information, which is replaced by meaningless numbers or letters. However, Backstrom et al. [1] pointed out that simply removing the identities of vertices could not preserve the privacy. Users can still be identified by attackers based on various background knowledge.

Many privacy preserving methods were proposed to defend against these attacks. Unlike the case in traditional relational datasets, privacy preserving in social network data publishing is a challenging problem:

- All the identities in the social network are connected with each other by edges, so any small modification may cause big changes to other vertices, sub-graph and even the whole network.
- It is very difficult to modify background knowledge because there are so much information can be used as background knowledge to re-identify the identities and breach the privacy.
- It is difficult to quantify information loss and utility. There are many elements in social networks, such as hubs, betweenness and communities, so we cannot simply compare two networks by vertices and edges. Additionally, utility is different based on different applications. We cannot use a unified standard to measure the utility.

Utility and privacy are contradicting elements. Most privacy preserving methods acquire a high level of privacy guarantee at the expense of utility. How to balance utility and privacy is a key problem when designing a privacy-preserving algorithm.

Our contributions in this paper are summarised as follows:

- We model the background knowledge that can be used by adversaries to break users' privacy.
- We classify the possible attack methods into two categories. One is that the adversary attempts to re-identify a specific person, and the other is that the adversary attempts to identify as many individuals as possible.
- We categorise the anonymization methods into two groups: anonymization and differential privacy. We review the privacy preserving models developed in recent 5 years.

The rest of this paper is organised as follows. We summarise the attack models in section 2. In section 3, we review the state-of-the-art privacy preserving methods from two categories: anonymization and differential privacy. Then we conclude the pa-

per and give the research direction in the future in section 4.

2 Attack Models

With the development of social network analysis and mining, privacy becomes an urgent problem that needs to be solved. While simply moving identifier is far from preserving the information, any background knowledge can be used by the adversary to attack the privacy easily.

2.1 Background Knowledge Attacker Utilizes

Background knowledge is the information that is known to adversaries, which can be used to infer privacy information of an individual in the social network. It plays an important role in modeling privacy attacks and developing countermeasures. We explore possible background knowledge that can be used by the adversary.

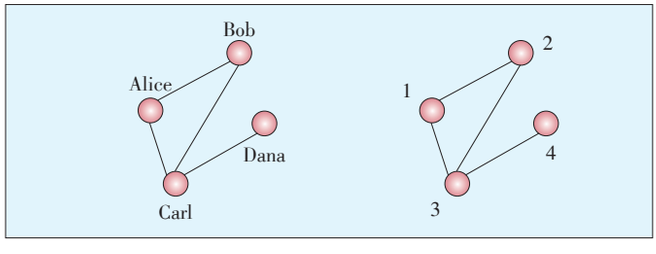
1) Vertices degree

Vertices degree represents how many direct connections between a node and its neighbours. Once the degree of the user is different from others in the graph, the vertex is re-identified. For example, in **Fig. 1**, node 3 and node 4 can be identified directly if the adversary knows Carl has three friends and Danna has only one friend.

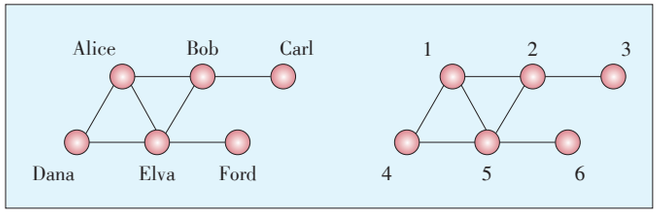
Tai et al. [2] identified a new attack called friendship attack, which is based on degree pair of an edge. They launched both degree and friendship attacks on 20Top-Conf dataset and proved that the friendship attack causes a much more privacy disclosure than the degree attack.

2) Neighbourhood

Neighbourhood refers to the neighbours of an individual who have connections with each other. Attackers make use of this kind of structural information to identify individuals [3], [4]. For example, in **Fig. 2**, if attackers know Bob has three friends and two neighbors and they connected with each other, Bob can be recognized in the anonymized graph.



▲**Figure 1. A degree attack.**



▲**Figure 2. A neighbourhood attack.**

Ninggal et al. [5] proposed another kind of attack called neighbourhood-pair attack, which uses a pair of neighbourhood structural information as background knowledge to identify victims. Such attacks assume attackers know more information than neighbourhood attacks do, so attackers have a higher chance to distinguish users in a dataset.

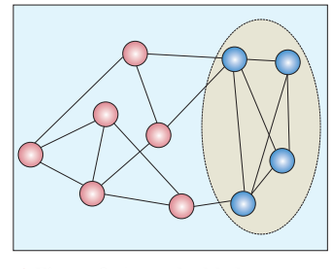
3) Embedded sub-graph

Sub-graph refers to a subset of the whole graph. Some adversaries create few fake nodes and build links using a specific way before the data is published, and then match the target graph with reference graph based on the sub-graph which has been planted. In **Fig. 3**, the grey part is the original graph, the black part is the sub-graph embedded by the adversary. Normally, the embedded sub-graph is unique and easy for attackers to identify after the dataset is released.

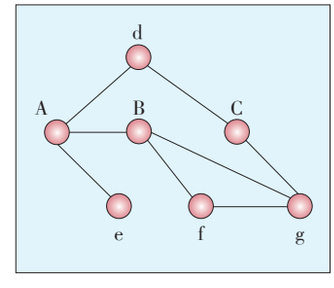
4) Link relationship

The relationship between two vertices also can be acquired by an adversary. Wang et al. [6] considered that the public users' identities are public and not sensitive. They utilized the connection between victims and public users to perform attack. For example, in **Fig. 4**, A, B, and C are public users, such as BBC and Michael Jackson. Their identities are publicity, and if attackers know vertex d has one hop to A and C and two hops to B, d can be identified.

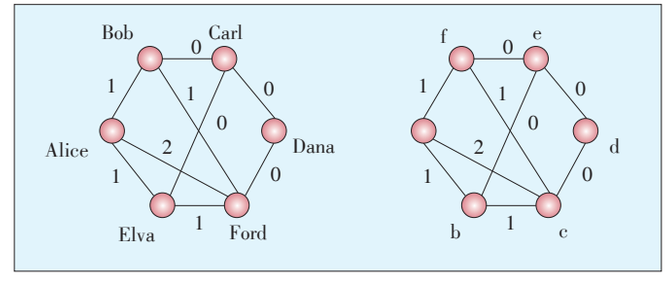
Sun et al. [7] committed a mutual friend attack. Their algorithm identifies a pair of users who connect to each other based on the number of mutual friends. For example, in **Fig. 5**, the numbers on the edge represent the number of mutual friends between two nodes. If the adversary knows Alice and Ford have two mutual friends, then she/he can identify a and c combined with other reference information (e.g. degree).



▲**Figure 3. An embedded sub-graph attack.**



▲**Figure 4. A fingerprint attack.**



▲**Figure 5. A mutual friends attack.**

5) Attributes of vertices

Attributes of individuals in a social network are represented as labels to vertices. Attackers may get the attributes of victims, such as age, sex and occupation. Such information can be helpful for adversaries to compromise users' privacy. For example, in Fig. 5, if the attacker knows Alice is a girl, Ford is a boy, he can identify them specifically by checking labeled sex information.

2.2 Attack Methods

There are two types of attacks. The first one is that the adversary tries to identify a specific person in the released dataset [5]–[7]. The other one is that the adversary attempts to identify as many individuals as possible. Most studies focus on the second one.

2.2.1 Structural Based Attacks

The first structural attack for re-identifying the vertices mutually present in two real and large social networks was proposed by Narayanan et al. [8]. They seeded 4-cliques and identified them by degrees and common neighbour counts. The propagation phase is based on the similarity score between the identified node and candidate node who has at least one mapped neighbour with the identified node. Their algorithm correctly identified 30.8% vertices just with 12.1% error. In their later work, Narayannan et al. [9] improved the seed identification process and formulated it as a combinatorial optimization problem, which improves the robustness compared with previous works.

Competing with Narayanan et al. [8], Peng et al. [10] proposed a two-stage de-anonymization algorithm. The attacker first plants a small specially designed sub-graph Gf, then propagate it based on two dissimilarity metrics. The experiment results showed their algorithm had a better efficiency even when the seed set was very small than Narayanan's algorithm. Besides, the incorrect identification number grows slowly when the initial seeds are growing. However the algorithm proposed by Peng et al. performed not well for large perturbation percentage. Besides, the test dataset used was too small with just hundreds of vertices.

Simon et al. [11] presented a Grasshopper algorithm. They selected the top degree nodes as seeds and introduced a weighting scheme based on the number of mappings in the nodes' neighbourhood and set convergence criteria. Their algorithm achieved a better result compared to [8] when the attacker has a rather noisy knowledge (lower similarity between two graphs), but does not always have a good result on different datasets.

Ji et al. [12] defined two vectors in terms of many structural attributes. They used these vectors to map nodes between two networks. Only 5–10 seed nodes are enough to de-anonymize. However the complexity and computational cost of this algorithm are very high.

There is another research group [13]–[16] maping the vertices between two similar graphs based on the graph matching algorithm. For example, Yartseva [13] introduced a simple percolation-based graph matching algorithm. This algorithm starts from a pre-matched seed set, and then maps the nodes with at least r neighbours that have been matched. In order to improve the precision, Korula et al. [15] matches the vertices from high degree to low degree according to the number of similarity witnesses. Chiasserini et al. [16] extended Yartseva's work [13] to a more realistic case that considering the power-low degree distribution. It makes the seed set as small as n.

2.2.2 Other Attack Methods

Most attacks are based on structure of the graph. However, there are other methods used to disclose users' privacy. Faresi et al. [17] and Goga et al. [18] used labeled attributes to correlate identical accounts on different social sites. Nilizadeh et al. [19] proposed a community-based de-anonymize method. They partitioned the graph into many communities. The algorithm identifies the seed communities first, and then maps the communities by creating a network of communities. Sharad et al. [20] proposed an automated de-anonymization method, which formulates the de-anonymizaiton problem as a learning task. The related research [21]–[27] focuses on predicting the link relationship, which can be used to disclose users' privacy as well.

3 Countermeasures

It is widely recognized that simply moving users' identity cannot guarantee their privacy. Many researchers pay much attention to this problem. We categorise the state-of-art privacy preserving methods into two categories: anonymization and differential privacy. **Table 1** shows the privacy models corresponding to attack models.

3.1 Anonymization

Anoymization is a popular method for preserving privacy, which is extensively used for social network data publishing.

▼Table 1. Privacy models

Privacy model	Attack model				
	Degree	Friendship	Neighbourhood	Sub-graph	Mutual friends
k-degree [29]	√				
structural diversity [34]	√				
k^2-degree-anonymity [2]		√			
k-neighbor [3]			√		
k-NMF-anonymity [7]					√
k-isomorphism [68]				√	
k-automorphism [69]	√		√	√	
differential Privacy [52]	√	√	√	√	√

We review these privacy preserving methods in this section. **Table 2** summarises recently developed anonymization methods from three aspects of privacy breach.

3.1.1 Preserving Vertices Identity

Most studies in recent years focus on preserving users' identity, preventing adversary re-identifying vertices in the graph. The main anonymization methods are based on k-anonymity [28], which means there are at least k nodes have the same structure with each other in the graph. It is realized by changing the structure of the original graph.

1) Graph Modification

Graph modification is a way that makes the graph satisfy k-anonymity by adding or deleting edges or nodes. State-of-art graph modification methods are summarised as follows.

In 2008, Liu and Terzi [29] first answered the question "how to minimally modify the graph to protect the identity of each individual involved?" They studied the vertices re-identity problem based on the degree background knowledge and provided a dynamic-programming algorithm for generating the k-anonymous graph based on the desired degree sequence.

Liu and Li [30] pointed out that the algorithm proposed in [29] had uncertainties. For example, if the anonymous graph construction process is random, the results will be totally different from the original graph. They developed two degree sequence partition algorithms. Those algorithms partition the degree sequence according to the partition cost calculated by the sum of difference of max neighbor vertices to their target degree. The nodes with smallest degree and distance are considered for constructing the graph satisfying k-degree anonymization. Noisy nodes are added when adding edges only cannot satisfy the constraint.

In order to guarantee the utility of anonymized graph, Wang et al. [31], [32] defined a measure standard Hierarchical Community Entropy (HCE) based on the hierarchical random graph (HRG) model to represent the information embedded in the graph community structure. They proposed an algorithm modifying edges that change the original graph to the nearest k-anonymizaiton graph.

Ninggal and Abawajy [33] introduced a utility-awared graph anonymizaiton algorithm. They use two metrics, Shortest Path Length (SPL) and Neighbourhood Overlap (NO) to quantify the utility. Compared to the scheme in [29], the algorithm in [33] introduces less distortion and improves utility preservation. However, this algorithm was only tested on four small datasets with hundreds of vertices. Besides, the computational cost of the algorithm is expensive.

Tai et al. [34] introduced a new privacy problem for protecting the community identity of vertices in the social network against degree attack. Even the graph satisfies k-degree anonymity, community identity still can be breached if the sets of nodes with the same degree belong to the same community. The authors proposed a structural diversity model by adding edges to make sure that the nodes with the same degree are distributed to at least k communities.

Tai et al. [2] introduced a Degree Sequence Anonymization algorithm to defend the friendship attack. The algorithm clusters vertices with similar degree, constructs at least k edges between two clusters by adding and deleting, and then adjusts the edges to k-anonimization under some conditions.

Zhou and Pei [3] provided a practical solution to defend neighborhood attack. They proposed a coding technique based on the depth-first search tree to represent neighborhood components. The algorithm tries to modify similar vertices as much as possible by adding edges to the vertices with the smallest degree, making sure every neighborhood sub-graph is isomorphic to at least k-1 other sub-graphs. But it does not consider the graph metric that may destroy the utility of the graph. In order to solve this problem, Okada et al. [35] extended the node selection function. They selected the closest node with the smallest degree and most similar label to suppress the changes

▼Table 2. Characters of anonymization algorithms

Anonymization algorithms	Operation	Information loss (anonymization cost)	Usability evaluation	Privacy disclosure		
				Vertices	attributes	links
[29]	EA, ESW	BD	GGP	√		
[30]	EA, VA	BD	GGP	√		
[31]	EA, ED, ESH	HCE	GGP	√		
[33]	EA, ED	UPM	GGP	√		
[34]	EA, VSP	BD	GGP	√		
[2]	EA, ED	BD	GGP	√		
[3]	EA, LG	BD	ANQ	√		
[35]	EA, LG	BD	GGP	√		
[7]	EA, ED	BD	GGP	√		
[36]	LG, EA, NA	BD	GGP, ANQ	√		
[4]	EA	BD	GGP	√	√	
[44]	EA, ED, VA	BD	GGP	√	√	
[45]	EA, VA	BD	GGP	√	√	
[46]	LG	BD	GGP	√	√	
[48]	REA	BD	GGP			√
[49]	ESH	–	LRP, RE, GGP			√
[50]	ESH	–	GGP			√
[40]	CL	SIL&DIL [51]	GGP	√		√
[38]	CL	SIL	SIL	√		√

ANQ: aggregate network queries
BD: based on distance
CL: clustering
DIL: descriptive information loss
EA: edge addition
ED: edge deletion
ESH: edge shifting
ESW: edge swapping
GGP: general graph properties
HCE: the change of Hierarchical Community Entropy value
LG: label generalization
LRP: link retention probability
PESW: possibility edge swapping
RE: reconstruction error
REA: random edge addition
SIL: structural information loss
UPM: The Utility Preserving Magnitude based on shortest path difference metric and neighborhood-overlap metric
VA: vertices addition
VD: vertices deletion
VSP: vertices splitting

of the distance of nodes. If the distance exceeds the threshold, the algorithm adds a noise node to suppress the changes of distance.

Wang et al. [36] considered the situation that attackers explore the sensitive information with labeled neighbourhood information as background knowledge. Their algorithm groups closest nodes according to the label sequence similarity. With different labels, each group contains at least one node. The authors modified the graphs in each group by label generalization, edge insertion and node insertion to make them isomorphic.

Sun et al. [7] proposed k-NMF anonymity to defend mutual friends attacks. This algorithm ensures that there exist at least k-1 other friend pairs in the graph that share the same number of mutual friends. The algorithm puts the edges into several groups and anonymizes each edge in the group one by one by adding edge. The algorithm chooses the candidate node that has maximum mutual friends with the vertex that need to be anonymized to ensure the utility of the graph, because the more mutual friends between the two vertices, the less impact the edge addition will have on the utility of the graph.

2) Clustering Methods

Clustering-based methods group the closest nodes together and show the original graph with super vertices and super edges. It shrinks the graph considerably, so it is not suitable for analysing local structure [37]. However it is a good method for answering aggressive queries.

Sihag [38] used clustering methods to make the vertices satisfy k-anonymization. They modeled the clustering process as an optimization problem and applied the genetic algorithm for choosing the best solution. The structural information loss proposed in [39] is used as the fitness function. A better solution is generated in each iteration until the terminating condition is satisfied.

Tassa and Cohen [40] introduced a sequential clustering algorithm to anonymise social network data. All nodes are clustered randomly to N/K groups (N is the vertices number, and K represents the cluster size). If Co is the cluster that node v belongs to, the information loss is calculated when moving v from Co to other clusters Ct. Node v is moved to the cluster that fits it best. This process is repeated until no vertices need to be moved to another cluster. This algorithm performs better in terms of reducing information loss and maintaining graph attributes than other clustering algorithms in [39] and [41]. In addition, the authors first applied the privacy preserving algorithm to a distributed social network.

3.1.2 Preserving Sensitive Attributes

The main method for preserving attributes in the social network is l-diversity [42], [43]. As an extension of the k-anonymity model, the l-diversity model reduces the granularity of data representation by using such techniques as generalization and suppression.

Paper [4], [44]–[46] all used l-diversity to protect the sensitive labels. Motivated by the observation that the nodes with high degree are usually famous people who have a relatively low privacy requirement, Jiao et al. [45] classified the nodes into three categories: High privacy requirement, middle and low, provided a personalized k-degree-l-diversity (PKDLD) model to protect nodes' degree and sensitive labels. Chen et al. [46] protected sensitive attributes in a weighted social network using l-diversity technology.

Rajaei et al. [47] provided ($\alpha, \beta, \gamma, \delta$) Social Network Privacy (SNP) to protect directed social network data with attributes against four types of privacy disclosure: presence, sensitive attribute, degree and relationship privacy. They grouped nodes with a high number of different properties and partition attributes to few tables and connected them by group IDs. This algorithm answers aggregate queries with high accuracy and maintains more data utility because the exact value is published and degree distribution is not changed. However, some false individuals would be generated during the process of anonymization, which may cause some errors.

3.1.3 Preserving Link

The basic technology for preserving link privacy is random perturbation. The main strategy is edge addition, deletion and switch.

Mittal et al. [48] proposed an algorithm preserving link privacy based on random walk. Their algorithm introduces fake edges with specific probability and defines a parameter t to control the noise that they want to add to the original graph.

Fard et al. [49] proposed a sub-graph-wise perturbation algorithm to limit link disclosure. They modeled the social network as a directed graph and partitioned vertices into some sub-graphs according to the closeness of nodes. The destination nodes are replaced by the nodes randomly selected from all destination nodes in the sub-graph with a certain probability. The algorithm preserves more graph structures compared with selecting from the whole graph. However, with the increasing of the number of sub-graphs, each sub-graph becomes very small, which increases the threats of identifying the link. In order to solve this drawback, neighbourhood randomization [50] was proposed. Selecting the destination nodes from the neighbourhood of the source node can avoid partition graph.

Ying and Wu [51] theoretically analysed how well the edge randomization approach protected the privacy of sensitive links, while Tassa and Cohen [40] believed that it is elusive and high non-uniform. Ying and Wu pointed out that some hub nodes with high degree are still distinguishable even when the algorithm has a high perturbation parameter. In addition, the random perturbation fails to provide a lower anonymization level (when k is small).

3.2 Differential Privacy

The main methods for protecting users' privacy are to modi-

fy the graph structure. Generally, these methods can only defend one specific kind of attacks and have no ability to resist the newly developed approaches. However, differential privacy [52] has been proved performing well in this direction.

Differential privacy is a mechanism that makes little difference to the results of the query with the addition or deletion of any tuple by adding random noise on the output. It works well on the tabular dataset preserving privacy. Some researchers also apply it to social networks [53]–[62], because it does not need to model background knowledge that is still a challenge for traditional anonymization methods. Besides, differential privacy is based on mathematics, which provides a quantitative assessment method andmakes the level of privacy protection comparable. We introduce it from two sides: node privacy and edge privacy.

3.2.1 Edge Privacy

Edge privacy makes negligible difference to the result of the query by adding or deleting a single edge between two individuals in the graph. The privacy dK-graph model [63] was used to enforce edge differential privacy [56]–[58]. The dK-series is used as the query function, but controllable noise is added based on the sensitivity parameter. In order to reduce the noise added to the dK-series, Sala et al. [57] provided a Divide randomize and Coonquer (DRC) algorithm, partitioning the data of dK-series into clusters with similar degree. It significantly reduces the sensitivity for each sub-series.

Wang and Wu [58] pointed out that Sala's approach was based on local sensitivity that may reveal information of the dataset (the example in [64]). Therefore, this approach could not achieve rigorous differential privacy. The authors in [58] used smooth sensitivity to calibrate the noise and achieved a strict differential privacy guarantee with smaller noise.

Xiao et al. [59] provided a novel sanitization solution that hides users' connection to others through differential privacy. They used the hierarchical random graph model (HRG) to infer the social network structure and record connection probabilities between all pair of vertices in the graph. In order to reduce the sensitivity, the Markov Chain Monte Carlo (MCMC) method is designed to sample a good HRG from the whole space. The sanitized graph is generated based on the identified HRG. This algorithm achieves a desirable utility due to smaller sensitivity compared with state-of-the-art works and effectively preserves some structural properties.

Edge privacy is a weaker guarantee than node privacy. Adversaries may still learn some general information. For example, high-degree nodes may have an identifiable effect on the query results [65]. However, it is practically strong enough in many applications, such as answering queries about individual relationship.

3.2.2 Node Privacy

Node privacy means adversaries do not have the ability to learn any information of an individual. It is very difficult to achieve node privacy while to guarantee the accurate query result, because the sensitivity is a very big result from adding or deleting nodes and connected edges. The query results would be too noisy to be applied in real life [66], [67], but it was proved a strong guarantee in some cases [65]. Some studies [54], [55] contributed to reduce sensitivity and returned accurate answers. However, existing algorithms cannot provide a good utility for practical applications. It is still an open problem.

4 Conclusions and Future Direction

In this paper, we first summarised and analysed the adversaries' attack methods to provide a good reference for researchers to design privacy preserving algorithms. Then we surveyed recently developed privacy preserving methods in two categories, anonymization and differential privacy. Though the privacy preserving methods are developed very well in the relational dataset, it is still in its infancy in social network datasets. For traditional method, there are few open problem need to be solved. First, define the information loss. The great majority of preserving methods do not have a specific definition of information loss. The number of edge and node addition and deletion is used to judge anonymizaiton cost, which is unreasonable. Second, defend against attacks with multiple types of background knowledge. If we want to develop traditional anonymization methods for privacy preserving, we need to consider that adversaries have various background knowledge, which is very practical in real life. Differential privacy can overcome some disadvantages of the traditional methods. For example, it does not based on any background knowledge and can quantify the level of privacy preserving as well. However, we cannot apply it directly, because the sensitivity of social networks is very high. How to reduce the sensitivity with less noise is a key research problem in the future.

References

[1] L. Backstrom, C. Dwork, and J. Kleinberg, "Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography," in *Proc. 16th International Conference on World Wide Web*, New York, USA, 2007, pp.181–190. doi: 10.1145/2043174.2043199.

[2] C.-H. Tai, P. S. Yu, D.-N. Yang, and M.-S. Chen, "Privacy-preserving social network publication against friendship attacks," in *Proc. 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, USA, 2011, pp. 1262–1270. doi: 10.1145/2020408.2020599.

[3] B. Zhou and J. Pei, "Preserving privacy in social networks against neighborhood attacks," in *IEEE 24th International Conference on Data Engineering (ICDE)*, Toronto, Canada, 2008, pp. 506–515. doi: 10.1007/s10115-010-0311-2.

[4] B. Zhang and J. Pei, "The k-anonymity and l-diversity approaches for privacy preservation in social networks against neighborhood attacks," *Knowledge and Information Systems*, vol. 28, no. 1, pp. 47–77, 2011. doi: 10.1007/s10115-010-0311-2.

[5] M. I. H. Ninggal and J. H. Abawajy, "Neighbourhood-pair attack in social network data publishing," in *Mobile and Ubiquitous Systems: Computing, Networking, and Services*, London, England, 2014, pp. 726–731. doi: 10.1007/978-3-319-11569-6_61.

[6] Y.Wang and B. Zheng, "Preserving privacy in social networks against connection fingerprint attacks," in *IEEE 31st International Conference on Data Engi-*

neering (ICDE), Seoul, Korea, 2015, pp. 54– 65. doi: 10.1109/ICDE.2015.7113272.
[7] C. Sun, P. S. Yu, X. Kong, and Y. Fu, "Privacy preserving social network publication against mutual friend attacks," in *IEEE 13th International Conference on Data Mining Workshops (ICDMW)*, Dallas, USA, 2013, pp. 883– 890. doi: 10.1145/1217299.1217302.
[8] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *30th IEEE Symposium on Security and Privacy*, Oakland, USA, 2009, pp. 173–187. doi: 10.1109/SP.2009.22.
[9] A. Narayanan, E. Shi, and B. I. Rubinstein, "Link prediction by de-anonymization: How we won the kaggle social network challenge," in *International Joint Conference on Neural Networks (IJCNN)*, San Jose, USA, 2011, pp. 1825–1834. doi: 10.1109/IJCNN.2011.6033446.
[10] W. Peng, F. Li, X. Zou, and J. Wu, "A two stage deanonymization attack against anonymized social networks," *IEEE Transactions on Computers*, vol. 63, no. 2, pp. 290–303, 2014. doi: 10.1109/TC. 2012.202.
[11] B. Simon, G. G. Guly_as, and S. Imre, "Analysis of grasshopper, a novel social network de-anonymization algorithm," *Periodica Polytechnica Electrical Engineering and Computer Science*, vol. 58, no. 4, pp. 161–173, 2014. doi: 10.3311/PPee.7878.
[12] S. Ji, W. Li, M. Srivatsa, J. S. He, and R. Beyah, "Structure based data de-anonymization of social networks and mobility traces," in *17th International Conference on Information Security*, Hong Kong, China, 2014, pp. 237– 254. doi: 10.1007/978-3-319-13257-0_14.
[13] L. Yartseva and M. Gross glauser, "On the performance of percolation graph matching," in *Proc. First ACM Conference on Online Social Networks*, Boston, USA, 2013, pp.119–130. doi: 10.1145/2512938.2512952.
[14] P. Pedarsani and M. Grossglauser, "On the privacy of anonymized networks," in *Proc. 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, USA, 2011, pp. 1235– 1243. doi: 10.1145/2020408.2020596.
[15] N. Korula and S. Lattanzi, "An efficient reconciliation algorithm for social networks," *Proc. VLDB Endowment*, vol. 7, no. 5, pp. 377– 388, 2014. doi: 10.14778/2732269.2732274.
[16] C. Chiasserini, M. Garetto, and E. Leonardi, "De-anonymizing scale-free social networks by percolation graph matching," in *INFORCOM*, Chicago, USA, 2015, pp. 1571–1579. doi: 10.1109/INFOCOM.2015.7218536.
[17] A. A. Faresi, A. Alazzawe, and A. Alazzawe, "Privacy leakage in health social networks," *Computational Intelligence*, vol. 30, no. 3, pp. 514–534, 2014. doi: 10.1111/coin.12005.
[18] O. Goga, H. Lei, S. H. K. Parthasarathi, G. Friedland, R. Sommer, and R. Teixeira, "Exploiting innocuous activity for correlating users across sites," in *Proc. 22nd International Conference on World Wide Web*, Rio de Janeiro, Brazil, 2013, pp. 447–458. doi: 10.1145/2488388.2488428.
[19] S. Nilizadeh, A. Kapadia, and Y.-Y. Ahn, "Community-enhanced de-anonymization of online social networks," in *Proc. ACM SIGSAC Conference on Computer and Communications Security*, Scottsdale, USA, 2014, pp. 537–548. doi: 10.1145/2660267.2660324.
[20] K. Sharad and G. Danezis, "An automated social graph de-anonymization technique," in *Proc. 13th Workshop on Privacy in the Electronic Society*, Scottsdale, USA, 2014, pp. 47–58. doi: 10.1145/2665943.2665960.
[21] L. Dong, Y. Li, H. Yin, H. Le, and M. Rui, "The algorithm of link prediction on social network," *Mathematical Problems in Engineering*, vol. 2013, article ID 125123, 2013. doi: 10.1155/2013/125123.
[22] N. Gupta and A. Singh, "A novel strategy for link prediction in social networks," in *Proc. 2014 CoNEXT on Student Workshop*, Sydney, Australia, 2014, pp. 12–14. doi: 10.1145/2680821.2680839.
[23] P. Sarkar, D. Chakrabarti, and M. Jordan. (2012). *Nonparametric link prediction in dynamic networks* [Online]. Available: http://arxiv.org/abs/1206.6394
[24] V. Malviya and G. P. Gupta, "Performance evaluation of similarity-based link prediction schemes for social network," in *1st International Conference on Next Generation Computing Technologies (NGCT)*, Dehradun, India, 2015, pp. 654–659. doi: 10.1109/NGCT. 2015.7375202.
[25] L. Duan, C. Aggarwal, S. Ma, R. Hu, and J. Huai, "Scaling up link prediction with ensembles," in *Proc. Ninth ACM International Conference on Web Search and Data Mining*, California, USA, 2016, pp. 367–376. doi: 10.1002/asi.v58:7.
[26] M. Al Hasan and M. J. Zaki, "A survey of link prediction in social networks," in *Social Network Data Analytics*, C. C. Aggarwal, Ed. Minneapolis, USA: Springer US, 2011, pp. 243–275. doi: 10.1007/978-1-4419-8462-3_9.
[27] Y. Dhote, N. Mishra, and S. Sharma, "Survey and analysis of temporal link prediction in online social networks," in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Mysore, India, 2013, pp. 1178–1183. doi: 10.1109/ICACCI. 2013.6637344.
[28] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression," SRI International Technical Report, Menlo Park, USA, Tech. Rep., 1998.
[29] K. Liu and E. Terzi, "Towards identity anonymizationon graphs," in *Proc. ACMSIGMOD International Conference on Management of Data*, Vancouver, Canada, 2008, pp. 93–106. doi: 10.1145/1117454.1117456.
[30] P. Liu and X. Li, "An improved privacy preserving algorithm for publishing social network data," in *IEEE International Conference on High Performance Computing and Communications & Embedded and Ubiquitous Computing*, Zhangjiajie, China, 2013, pp. 888 – 895. doi: 10.1109/HPCC.and.EUC. 2013.127.
[31] Y. Wang, L.Xie, B. Zheng, and K. C. Lee, "High utility k-anonymization for social network publishing," *Knowledge and Information Systems*, vol. 41, no. 3, pp. 697–725, 2014. doi: 10.1007/s10115-013-0674-2.
[32] Y. Wang, L. Xie, B. Zheng, and K. C. Lee, "Utility-oriented k-anonymization on social networks," in *16th International Conference on Database Systems for Advanced Applications*, Hong Kong, China, 2011, pp. 78–92.
[33] M. I. H. Ninggal and J. H. Abawajy, "Utility-aware social network graph anonymization," *Journal of Network and Computer Applications*, vol. 56, pp. 137–148, 2015. doi: 10.1016/j.jnca.2015.05.013.
[34] C.-H. Tai, S. Y. Philip, D.-N. Yang, and M.-S. Chen, "Structural diversity for privacy in publishing social networks," in *SIAM International Conference on Data Mining*, Mesa, USA, 2011, pp. 35–46. doi: 10.1137/1.9781611972818.4.
[35] R. Okada, C. Watanabe, and H. Kitagawa, "A k-anonymization algorithm on social network data that reduces distances between nodes," in *IEEE 33rd International Symposium on Reliable Distributed Systems Workshops (SRDSW)*, Nara, Japan, 2014, pp. 76–81. doi: 10.1109/SRDSW.2014.19.
[36] Y. Wang, F. Qiu, F. Wu, and G. Chen, "Resisting label-neighborhood attacks in outsourced social networks," in *Performance Computing and Communications Conference (IPCCC)*, Austin, USA, 2014, pp. 1–8. doi: 10.1109/PCCC. 2014.7017106.
[37] B. Zhou, J. Pei, and W. Luk, "A brief survey on anonymization techniques for privacy preserving publishing of social network data," *ACM Sigkdd Explorations Newsletter*, vol. 10, no. 2, pp. 12– 22, 2008. doi: 10.1145/1540276.1540279.
[38] V. K. Sihag, "A clustering approach for structural k-anonymity in social networks using genetic algorithm," in *Proc. CUBE International Information Technology Conference*, Pune, India, 2012, pp. 701– 706. doi: 10.1145/2381716.2381850.
[39] A. Campan and T. M. Truta, "Data and structural k-anonymity in social networks," in *Second ACM SIGKDD International Workshop PinKDD*, Las Vegas, USA, 2009, pp. 33–54. doi: 10.1007/978-3-642-01718-6_4.
[40] T. Tassa and D. J. Cohen, "Anonymization of centralized and distributed social networks by sequential clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 2, pp. 311–324, 2013. doi: 10.1109/TKDE.2011.232.
[41] E. Zheleva and L. Getoor, "Preserving the privacy of sensitive relationships in graph data," in *1st ACM SIGKDD International Conference on Privacy, Security, And Trust in KDD*, San Jose, USA, 2008, pp. 153– 171. doi: 10.1145/1117454.1117456.
[42] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," in *Proc. 22nd IEEE International Conference on Data Engineering (ICDE)*, Washington, USA, 2006, pp. 24–24. doi: 10.1109/2006. doi: 10.1109/ICDE.2006.1.
[43] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, 2007. doi: 10.1145/1217299.1217302.
[44] M. Yuan, L. Chen, P. S. Yu, and T. Yu, "Protecting sensitive labels in social network data anonymization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 3, pp. 633–647, 2013. doi: 10.1109/TKDE.2011.259.
[45] J. Jiao, P. Liu, and X. Li, "A personalized privacy preserving method for publishing social network data," in *11th Annual Conference on Theory and Applications of Models of Computation*, Chennai, India, 2014, pp. 141– 157. doi: 10.1007/978-3-319-06089-7_10.
[46] K. Chen, H. Zhang, B. Wang, and X. Yang, "Protecting sensitive labels in weighted social networks," in *Web Information System and Application Conference (WISA)*, Yangzhou, China, 2013, pp. 221– 226. doi: 10.1109/WISA.2013.50.
[47] M. Rajaei, M. S. Haghjoo, and E. K. Miyaneh, "Ambiguity in social network data for presence, sensitive attribute, degree and relationship privacy protection," *PLOS ONE*, vol. 10, no. 6, 2015. doi: 10.1371/journal.pone.0130693.
[48] P. Mittal, C. Papamanthou, and D. Song. (2012). *Preserving link privacy in so-*

cial network based systems [Online]. Available: http://arxiv.org/abs/1208.6189
[49] A. M. Fard, K. Wang, and P. S. Yu, "Limiting link disclosure in social network analysis through sub graph-wise perturbation," in *Proc. 15th International Conference on Extending Database Technology*, Berlin, Germany, 2012, pp. 109–119. doi: 10.1109/ICDE.2011.5767905.
[50] A. M. Fard and K. Wang, "Neighborhood randomization for link privacy in social network analysis," *World Wide Web*, vol. 18, no. 1, pp. 9–32, 2015. doi: 10.1007/s11280-013-0240-6.
[51] X. Ying and X. Wu, "On link privacy in randomizing social networks," *Knowledge and Information Systems*, vol. 28, no. 3, pp. 645–663, 2011.
[52] Dwork, "Differential privacy," in *International Colloquium on Automata, Languages and Programming*, Venice, Italy, 2006, pp. 1–12.
[53] J. Blocki, A. Blum, A. Datta, and O. Shefet, "Differentially private data analysis of social networks via restricted sensitivity," in *Proc. 4th Conference on Innovations in Theoretical Computer Science*, Berkeley, USA, 2013, pp. 87–96. doi: 10.1145/2422436.2422449.
[54] S. Chen and S. Zhou, "Recursive mechanism: towards node differential privacy and unrestricted joins," in *Proc. ACM SIGMOD International Conference on Management of Data*, New York, USA, 2013, pp. 653–664. doi: 10.1145/2463676.2465304.
[55] S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith, "Analyzing graphs with node differential privacy," in *Theory of Cryptography*, Tokyo, Japan, 2013, pp. 457–476. doi: 10.1007/978-3-642-36594-2_26.
[56] D. Proserpio, S. Goldberg, and F. McSherry, "A work flow for differentially-private graph synthesis," in *Proc. ACM Workshop on Online Social Networks*, Helsinki, Finland, 2012, pp. 13–18. doi: 10.1145/2342549.2342553.
[57] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao, "Sharing graphs using differentially private graph models," in *Proc. ACM SIGCOMM Conference on Internet Measurement Conference*, Berlin, Germany, 2011, pp. 81–98. doi: 10.1007/s00778-006-0039-5.
[58] Y. Wang and X. Wu, "Preserving differential privacy in degree-correlation based graph generation," *Transactions on Data Privacy*, vol. 6, no. 2, pp. 127–145, Aug. 2013. doi: 10.1145/1866739.1866758.
[59] Q. Xiao, R. Chen, and K.-L. Tan, "Differentially private network data release via structural inference," in *Proc. 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, USA, 2014, pp. 911–920. doi: 10.1007/s00778-013-0344-8.
[60] M. Kapralov and K. Talwar, "On differentially private low rank approximation," in *Proc. 24h Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, USA, 2013, pp. 1395–1414. doi: 10.1137/1.9781611973105.101.
[61] Y. Wang, X. Wu, and L. Wu, "Differential privacy preserving spectral graph analysis," in *17th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Gold Coast, Australia, 2013, pp. 329–340. doi: 10.1007/978-3-642-37456-2_28.
[62] F. Ahmed, R. Jin, and A. X. Liu. (2013). *A random matrix approach to differential privacy and structure preserved social network graph publishing* [Online]. Available: http://arxiv.org/abs/1307.0475
[63] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat, "Systematic topology analysis and generation using degree correlations," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 135–146, 2006. doi: 10.1145/1159913.1159930.
[64] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proc. Thirty-Ninth Annual ACM Symposium on Theory of Computing*, San Diego, USA, 2007, pp. 75–84. doi: 10.1145/1250790.1250803.
[65] C. Task and C. Clifton, "What should we protect? defining differential privacy for social network analysis," in *State of the Art Applications of Social Network Analysis*, Springer, 2014, pp. 139–161. doi: 10.1007/978-3-319-05912-9_7.
[66] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Ninth IEEE International Conference on Data Mining (ICDM)*, Miami, USA, 2009, pp. 169–178. doi: 10.1109/ICDM.2009.11.
[67] D. Kifer and A. Machanavajjhala, "No free lunch in data privacy," in *Proc. ACM SIGMOD International Conference on Management of Data*, Bangalore, India, 2011, pp. 193–204. doi: 10.1145/1217299.1217301.
[68] J. Cheng, A. W.-C. Fu, and J. Liu, "K-isomorphism: privacy preserving network publication against structural attacks," in *Proc. ACMSIGMOD International Conference on Management of Data*, Indianapolis, USA, 2010, pp. 459–470. doi: 10.1145/1807167.1807218.
[69] L. Zou, L. Chen, and M. T.Ozsu, "K-automorphism: a general framework for privacy preserving network publication," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 946–957, 2009. doi: 10.14778/1687627.1687734.

Manuscript received: 2016-02-17

Biographies

YANG Mengmeng (ymengm@deakin.edu.au) received her BE from Qingdao Agricultural University, China in 2007, and M.Eng from Shenyang Normal University, China in 2014. She is currently a PhD candidate in the School of Information Technology, Deakin University, Australia. Her research interests include privacy preserving, network security and machine learning.

ZHU Tianqing (t.zhu@deakin.edu.au) received her BE and ME degrees fromWuhan University, China, in 2000 and 2004, respectively, and a PhD degree from Deakin University in Computer Science, Australia, in 2014. She is currently a continuing teaching scholar in the School of Information Technology, Deakin University, Australia. Her research interests include privacy preserving, data mining and network security. She has won the best student paper award in PAKDD 2014.

ZHOU Wanlei (wanlei.zhou@deakin.edu.au) received his BE and ME degrees from Harbin Institute of Technology, China in 1982 and 1984, respectively, and a PhD degree from The Australian National University, Australia, in 1991, all in Computer Science and Engineering. He also received a DSc degree from Deakin University in 2002. He is currently the Alfred Deakin Professor and Chair Professor in Information Technology, School of Information Technology, Deakin University. His research interests include distributed systems, network security, bioinformatics, and e-learning. Professor Zhou has published more than 300 papers in refereed international journals and refereed international conferences proceedings, including over 30 articles in IEEE journal in the last 5 years.

XIANG Yang (yang.xiang@deakin.edu.au) received his PhD in Computer Science from Deakin University, Australia. He is the Director of Centre for Cyber Security Research, Deakin University. He is the Chief Investigator of several projects in network and system security, funded by the Australian Research Council (ARC). His research interests include network and system security, data analytics, distributed systems, and networking. He has published more than 200 research papers in many international journals and conferences. Two of his papers were selected as the featured articles in the April 2009 and the July 2013 issues of *IEEE Transactions on Parallel and Distributed Systems*. Two of his papers were selected as the featured articles in the Jul/Aug 2014 and the Nov/Dec 2014 issues of *IEEE Transactions on Dependable and Secure Computing*.

DOI: 10.3969/j. issn. 1673-5188. 2016. S0. 002
http://www.cnki.net/kcms/detail/34.1294.TN.20160607.1841.002.html, published online June 7, 2016

Verification of Substring Searches on the Untrusted Cloud

Faizal Riaz-ud-Din and Robin Doss
(School of Information Technology, Deakin University, Burwood, VIC 3125, Australia)

Abstract

Ensuring the correctness of answers to substring queries has not been a concern for consumers working within the traditional confines of their own organisational infrastructure. This is due to the fact that organisations generally trust their handling of their own data hosted on their own servers and networks. With cloud computing however, where both data and processing are delegated to unknown servers, guarantees of the correctness of queries need to be available. The verification of the results of substring searches has not been given much focus to date within the wider scope of data and query verification. We present a verification scheme for existential substring searches on text files, which is the first of its kind to satisfy the desired properties of authenticity, completeness, and freshness. The scheme is based on suffix arrays, Merkle hash trees and cryptographic hashes to provide strong guarantees of correctness for the consumer, even in fully untrusted environments. We provide a description of our scheme, along with the results of experiments conducted on a fully-working prototype.

Keywords

substring search; query verification; cloud

1 Introduction

The paradigm shift from traditional, locally-hosted databases and infrastructure to their deployment on the cloud has provided a robust solution for those seeking to minimising costs whilst at the same time greatly enhancing flexibility. However, in spite of these benefits there are a number of areas of concern over the control of the data that gets outsourced, as well as the question of trust that arises when one hands over data and processing to a cloud service provider (CSP).

An elegant solution to this trust problem would be reducing requirements of trust in the relationship between the user and the CSP, and instead verifying the computation performed by the CSP and the authenticity of the data received by the user. This approach, known as query and data verification, attempts to provide data processing guarantees to consumers that cannot be falsified. Such schemes require the server to return a proof of correctness (known as a verification object or VO) with the results, which is used by the client to verify the correctness of the results.

This paper focuses on a substring query verification scheme for string matching queries against file-based data hosted on untrusted cloud servers. Substring queries match arbitrary substrings to larger strings. Research in the area of substring query verification has been scarce, with keyword search verification being more prevalent.

1.1 Our contributions

Our contributions in this paper may be summarised as follows:

- To the best of our knowledge, we provide the first existential substring query verification scheme that satisfies the properties of completeness, authenticity, and freshness.
- We show that our scheme detects both false positive and false negative query results, as opposed to the closest comparable substring matching verification scheme [1] that provides detection of false positives, but fails to provide proof for the detection of false negatives.
- Our scheme provides smaller VO sizes than the closest comparable substring matching verification scheme for large matches.

1.2 Motivation

There are commonly three types of substring searches that are executed on strings: existential, counting and enumeration queries. Typically, a string x is sought in a string S. Existential queries test whether x exists in S, counting queries that return the number of occurrences of x in S, and enumeration queries list all of the positions in S where x occurs. We focus on providing verification for existential queries in this paper, which we may also refer to simply as substring searches.

Proper substring searches provide greater flexibility and control with what is being searched for than keyword searches (where the search focuses on whole words rather than partial words). As such, one may be able to search for partial words, a combination of words where the text being matched begins or ends in the middle of a word, or in blocks of text that are composed entirely of characters without separators. Suffix trees, suffix arrays, finite automaton, and other techniques are used for realising proper substring searches. We focus on providing a verification scheme for these types of substring searches.

In the cloud environment, without a query verification mechanism in place, the only guarantee clients have of receiving correct answers to queries against cloud-hosted data is the trust between them and the CSP. However this may not always be sufficient, and definitely does not provide an absolute guarantee of correct query results.

1.3 Applications

Providing verification that a query has been correctly executed and that the received response with respect to the submitted query is correct is of tantamount importance for applications running on the cloud. With respect to existential query verification the following represents a small subset of applications that would benefit from our scheme:

- Querying large sets of biological data for specific occurrences of smaller DNA or RNA sequences as is required in sequence alignment algorithms. Our scheme may be used as a building block upon which to construct sequence alignment algorithms that provide proofs of correctness for alignment queries executed on remote cloud servers.
- Querying databases for partial matches of registration numbers, which may consist of alphanumeric and special characters. Our scheme can provide a basis upon which pattern matching queries may be verified for correctness when executed against databases stored on the cloud.
- Proving guarantees of correctness for queries issued against sensitive data such as medical records that may be stored on remote servers.
- Verification of answers to queries against text data created using agglutinative languages [2] where distinguishable words are not well-defined. Although languages such as English, where words are well-defined, may benefit from inverted indexes based on terms, and therefore from verification schemes based on inverted indexes, agglutinative languages may not fully benefit from such indexes, and our scheme provides a more robust method for providing search result verification against searches on such languages.

2 Related Work

Although the research in the areas of file verification at the block-level and byte level is plentiful [3]–[6], there has been relatively little work done in the area of existential substring query verification against single or multiple files.

The same can be said for substring query verification in the area of databases. Research into providing verification for queries based on numeric-based predicates [7]–[10] is plentiful with a number of papers published recently. A number of schemes based on Merkle hash trees (see Subsection 3.3) [7], [11]–[14], signature-chaining [15]–[17], and other approaches [18]–[22] have been presented in the literature largely with respect to numeric-based query verification.

However, substring queries look at a portion of the value in a given tuple attribute. Since any combination of the characters making up the value could be searched for, it is harder to find efficient verification schemes.

The work presented in [23], [24] provides substrings search verification. However, the study is based on inverted indexes, which are limited to providing keyword-level verification. As such, although they provide verification for substring searches within documents, they do so at a higher granularity than what is achieved by proper substring searches.

The scheme most closely related to ours is presented by Martel et al. in [1]. They propose a model called a Search Directed Acyclic Graph (DAG), which is used to provide methods to compute VO for a number of different data structures. One of these data structures is the suffix tree that is used to provide verification of proper substring searches. Their verification scheme uses hashing and techniques similar to that of Merkle Hash Trees to achieve verification of substring searches. However, although they provide proofs for detecting false positive queries, they do not do so for false negative queries. Additionally, their scheme, although efficient for small substring searches, would incur large bandwidth costs for longer substring searches. We address these two concerns and propose our method that also provides authenticity, completeness, and freshness.

3 Preliminaries

In this section we briefly describe the cryptographic primitives we use in our proposed scheme.

3.1 Secure Hash Function

A secure hash function, $h(x) \rightarrow d$, takes as input an arbitrary-length string, x, and produces a fixed-length hash digest, d. The one-way property of the secure hash function guarantees that given only a hash digest, d, it is infeasible to produce the original input string, x. The collision-free property of the secure hash function guarantees that given two distinct strings, x and y, their respective hash digests, d and e, will never be the same i.e. $h(x) \neq h(y)$. Commonly used secure hash schemes are MD5 [25], SHA1 [26] and SHA2 [27].

3.2 Digital Signature

A digital signature scheme consists of key generation, sign-

ing, and verification algorithms. The key generation algorithm produces a pair of related keys, known as public (pk) and private, or secret (sk) keys. The signing algorithm $sign(sk, m) \rightarrow m_{sk}$ takes as input the private key (sk) and a message (m) to produce a digital signature (m_{sk}). The verification algorithm $verify(pk,m_{sk},m_r) \rightarrow (Y|N)$ takes as input the signature m_{sk}, the public key pk, and the received message m_r, and returns a Y or N to either affirm correctness of the received message m_r with respect to the original message m or to deny it. The most commonly used digital signature scheme is RSA [28].

3.3 Merkle Hash Tree

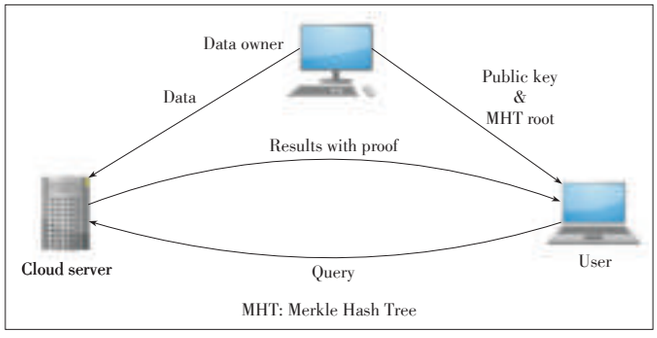
A Merkle Hash Tree (MHT) [29] is a binary tree that has as its leaves secure hash digests of data items. Each node in the tree is formed by concatenating the hashes of its child nodes, and then hashing the concatenated hashes. The root hash is signed with the owner's private key, and then can be validated using the owner's public key. The MHT allows verification of the order of the data items, their individual values, and a range of values. Verification involves retrieving the specified value, and then finding all nodes that are siblings of all nodes from the specified leaf to the root. The given data value is then hashed and combined with the other node hashes to regenerate the root, which is then checked against the original root to confirm or reject the value. MHT may also be implemented as B+ trees to improve efficiency.

4 Proposed Scheme

In this section, we describe our proposed scheme for the verification of substring queries.

4.1 System Model

A typical system model for our proposed substring query verification scheme is illustrated in **Fig. 1**. The data owner holds data that is queried by users. However, due to resource constraints or other reasons, the data owner wishes to outsource the data and query processing to the cloud for easier management. The cloud server therefore becomes a proxy for hosting data and query processing on behalf of the data owner.



▲**Figure 1. System model for the proposed substring query verification scheme.**

The data owner pre-processes the data, the final step of which includes generating an MHT root. The owner stores the root of the MHT and publishes the string data to the server, and publishes the public key within the context of a public key infrastructure.

A user submits substring queries to the server, which in response executes the query to obtain the query result. The query result is then sent back to the user together with a VO. The user then processes the query result and VO with a verification algorithm to either accept or reject the result.

4.2 Notation

Table 1 shows the definitions of pre-defined operations that are used in the rest of the paper. **Table 2** provides a list of the notations used in this paper to facilitate the descriptions of our proposed scheme.

4.3 Verification Properties

Verification schemes must provide solutions for at least three desirable properties authenticity, completeness, and freshness. These properties have been described in the literature previously [7], [16], [18], [30], but for the purpose of our scheme, we provide definitions of these properties below in the context of existential queries.

Consider a result R of an existential query Q that has been executed fully and correctly, in an uncompromised environment, on a string S that comprises substrings $\{S_1 \dots S_n\}$. Over time, S undergoes updates, causing its state to change from S^0, signifying the initial state at time 0, through to the current state at the current time, signified by S^c. The qualifying substrings in S that satisfy the predicates of Q are denoted $\{S^1_Q \dots S^1_m\}$. We denote the existential function as F. We call

▼**Table 1. Pre-defined operations and their descriptions**

Operation	Definition
$BuildMHT(v_1,\cdots,v_n) \rightarrow mht$	Builds an MHT based on data values sorted in a specified order from v_1 to v_n. Outputs a new MHT, mht
$BuildSA(S) \rightarrow sa$	Builds a suffix array from string S. Outputs a new suffix array, sa.
$BuildVO(V,N) \rightarrow VO$	Builds a VO from a set of values (V), and a set of MHT nodes (N). Outputs a new VO
$IsSubstringOf(x,S,sa) \rightarrow <\{Y,N\}, i, \hat{i}_l, \hat{i}_r>$	Checks whether string x is a substring of string S using S's suffix array, sa. Outputs Y if x is a substring of S, otherwise outputs N. If x is a substring of S, sets i to the first index in sa where x prefixes $S_{suff(sa_i)}$ Otherwise, if x is not a substring of S, $\hat{i}_l$ and $\hat{i}_r$ are set to the neighbouring indexes in sa between which x would have been found, had it existed in sa.
$GetSinglePathSiblings(v,mht) \rightarrow N$	Traverses the given mht from the leaf v to the root, adding all siblings along that path to the set N
$GetRangePathSiblings(v_l,v_r,mht) \rightarrow N$	Traverses the given mht from each of the leaves v_l and v_r to the root, adding all siblings along the paths that are not ancestors of either leaf to the set N

▼Table 2. Notations used in this paper

Notation	Definition	Notation	Definition
DO	Data owner	x	Substring being sought
U	Data user	R	Root of an MHT
CS	Cloud server	R_{DO}	Root signed with DO's secret key
S	String	R_{DO}	DO's secret key
$\|S\|$	Size of string S	R_{DO}	DO's public key
S_i	Character at position i in S	L_i	Leaf I of an MHT
$S_{i\cdots j}$	Substring of S from S_i to S_j	VO	Verification object
$S_{suff(i)}$	Suffix of S starting at i	Q	A query
sa_S	Suffix array of S	QR	A query result
$sa(i)$	Element i in sa		

R the correct, uncompromised, unaltered result of Q. In the case where Q is issued remotely to D, which resides in a possibly untrusted environment, we consider the authenticity, completeness and freshness of the final result R' received by a remote user as follows.

Definition 4.1: Authenticity in our scheme means that R' is a result of executing F only on the substrings of the uncompromised string (i.e. that was created by the data owner). Specifically, authenticity of an existential query result R' is satisfied when

$$R' = F\left(\left\{S'_1 \cdots S'_k\right\}\right) \wedge S'_i \in \left\{S'_1 \cdots S'_k\right\} : S'_i \in (S^0 \cup \cdots \cup S^C).$$

Definition 4.2: Completeness in our scheme means that R' is a result of executing F on the same number of substrings as that executed by a correctly executing Q on an uncompromised string. Specifically, completeness of an existential query result R' is satisfied when

$$R' = F\left(\left\{S'_1 \cdots S'_k\right\}\right) \wedge \left|\left\{S'_1 \cdots S'_k\right\}\right| = \left|\left\{S^Q_1 \cdots S^Q_n\right\}\right|.$$

Definition4.3: Freshness in our scheme means that R' is a result of executing F on the most recently updated version of the uncompromised string. Specifically, freshness of an existential query result R' is satisfied when $R' = F\left(\left\{S'_1 \cdots S'_k\right\}\right) \wedge \left|\left\{S'_1 \cdots S'_k\right\}\right| \in \left\{S'_1 \cdots S'_n\right\} : S'_i \in S^C$.

4.4 Suffix Arrays

Let S be a string composed of characters from a set Σ of fixed sized, finite ordered alphabets. The length of S is denoted by n. \$ specifies a special end-of-string marker, which is smaller than all alphabets in Σ, but which does not occur in S. $S[i]$ denotes the index of the ith character in S.

The suffix array sa of the string S is an array of length $n+1$, where the elements in the array are unique indexes in $S \mid \$$, and are ordered lexicographically based on the suffixes of S, where each element points to a different suffix as indicated by its indexing value.

Table 3 shows an example of the suffix array for the string *'aardvark'*. The end-of-string marker (\$) is appended to the string before the suffix array is constructed, and has the smallest value out of all the alphabets in the string, resulting in its being sorted to the first element. The remaining suffixes' indexes are placed in lexicographical order in the suffix array.

4.5 Assumptions

We assume the following for the correctness of our scheme:

1) DO's public key has been obtained by U, possibly through a secure channel or reliable public-key infrastructure.
2) DO's secret key has not been compromised.
3) The public key encryption scheme used is secure under appropriately specified parameters.
4) It is infeasible to find collisions in the secure hash function that is used as a basis for the hash chaining and MHT generation procedures.

4.6 Hash Chains with Sequential Indexing

We introduce the concept of hash chains with sequential indexing (HCSI) as a building block for our scheme. The HCSI is essentially a hash chain with each link on the hash chain being tagged with a sequential identifier.

Definition 4.4: The HCSI is defined recursively as follows:

$$HCSI(S_i) = \begin{cases} h\left(i \,//\, S_i\right) & if\ i = |S| \\ h\left(i \,//\, S_i \,//\, HCSI(S_{i+1})\right) & if\ i < |S| \\ null & if\ i > |S| \end{cases}$$

The HCSI allows the specification of four variables that are useful for our scheme $\alpha, \beta, S_\beta, \gamma$, where α and β are indexes in S, S_β is the character at index β in S, and γ is a hash digest.

Definition 4.5: $\bar{x}$ is a prefix of $S_{suff(sa(i))}$ that is minimally unmatched to x, s.t. $\bar{x}_{0\cdots k-1} = x_{0\cdots k-1} \wedge \bar{x}_k \neq x_k \wedge k \in \{0\cdots|x|\}$.

Definition 4.6: α refers to the position at the head of the HCSI, and is defined in the context of the suffix that it is associated with, as follows: $\alpha(S_{suff(i)}) = i$. In other words, α is the position in the HCSI that corresponds to the first character in $S_{suff(i)}$, and has the same value as $sa(i)$.

▼Table 3. Suffix array for the string '*aardvark*'

SA index	SA value	Resulting suffix
0	9	\$
1	1	aardvark \$
2	2	ardvark \$
3	6	ark \$
4	4	dvark \$
5	8	k \$
6	3	rdvark \$
7	7	rk \$
8	5	vark\$

Definition 4.7: β refers to the position of the last matching character in $S_{suff(i)}$ when x is a prefix of $S_{suff(i)}$, otherwise it refers to the first non-matching character in $S_{suff(i)}$ if x is not a prefix of $S_{suff(i)}$. In the case where x prefixes $S_{suff(i)}$, $\beta = i + |x| - 1$, otherwise if x does not prefix $S_{suff(i)}$, $\beta = i + |\bar{x}| - 1$.

Definition 4.8: S_β is the character in S at position β. If x is a prefix of $S_{suff(i)}$, $S_\beta = x_{|x|}$, otherwise $S_\beta = \bar{x}_{\bar{x}}$

Definition 4.9: γ refers to the first hash digest in the HCSI occurring after β, and is defined as $\gamma(\beta) = HCSI(S_{\beta+1})$.

By utilising the HCSI variables as given above, we are able to minimise the hash operations performed by the user from $|S_{suff(i)}| + 1$ hashes to at most $|x| + 1$ hashes, where $|x| < |S_{suff(i)}|$.

The sequential identifiers allow us to reduce the number of hashing operations to be performed on the user's end, and also reduces the communication cost by sending to the user only those hashes in the chain that the user needs to perform verification. **Algorithm 1** shows how the reconstruction may be realised.

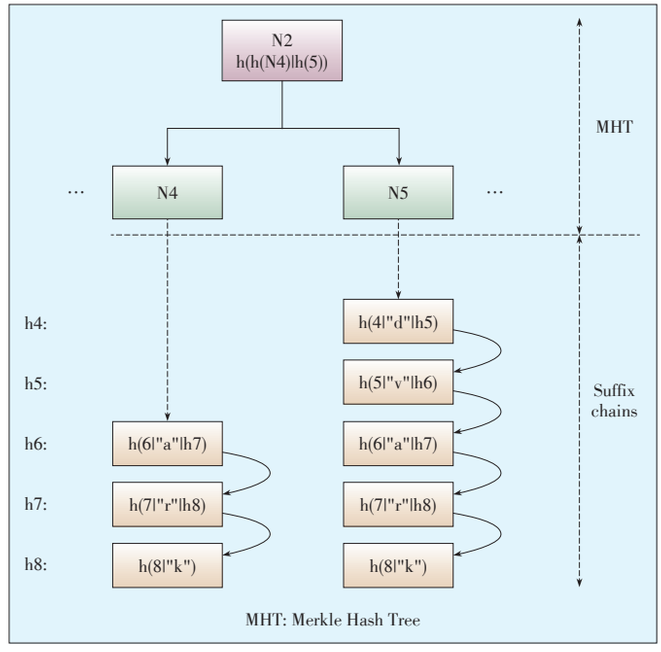
Algorithm 1: HCSI Reconstruction

Input: $x, \alpha, \beta, S_\beta, \gamma$
if $!(S_\beta)$ **then**
  /* Matching reconstruction */
    $\beta' \leftarrow \alpha + |x| - 1$; $S'_\beta \leftarrow x_{|x|}$; $x' \leftarrow x$;
else
  if $(\beta \ \alpha + 1 \leq |x|)\ (x_{\beta-\alpha+1} \neq S_\beta)$ **then**
    /* Non-matching reconstruction */
    $\beta' \leftarrow \beta$; $S'_\beta \leftarrow S_\beta$; $x' \leftarrow x_{1\cdots\beta-\alpha} // S_\beta$;
  else
    /* Invalid β and/or S_β */
    reject;
  end
end
$hcsi \leftarrow h(\beta', S'_\beta, \gamma)$;
for $i \leftarrow \beta' - 1$ **down-to** α **do**
  $hcsi \leftarrow h\left(i, x'_{i-\alpha+1}, hcsi\right)$;
end

4.7 Scheme Outline

Our intuition is to leverage MHTs to act as verification structures for suffix arrays. In essence, we build an MHT on top of a suffix array (**Fig. 2**), and then allow query results to be passed to the user along with VOs as proofs for the result. The user then verifies the result using the VO.

We define five phases for the implementation of our scheme: Setup Phase, Query Phase, Query Response Phase, Verification Phase, and Update Phase.



▲**Figure 2. An example of the sequenced hash chain of the suffixes '*ark*' and '*dvark*', and their corresponding leaves in a partially illustrated MHT.**

1) Phase 1: Setup

Data owners initiate the scheme by firstly generating a suffix array from the string or text file that their wishes to make available for querying. They then build HCSI digests over the suffixes in the string. The HCSI digests for each suffix are then ordered according to the suffix array indexes and an MHT is constructed over them. The data owners then sign the MHT root with their private keys and upload the string to the server. They also transmit their public keys to the users. They may optionally discard the string, the suffix array, and the MHT. **Algorithm 2** illustrates this phase.

Algorithm 2: Setup

Input: S, sk_{DO}
$sa \leftarrow BuildSA(S)$;
foreach $sfx \leftarrow \left(S_{suff(sa_S(1))} \cdots S_{suff(sa_S(|S|))}\right)$ **do**
  $v_i \leftarrow HCSI(sfx)$;
end
$mht \leftarrow BuildMHT(v1, ..., vn)$;
$R_{DO} \leftarrow \text{sign}(sk_{DO}, R)$;
$Upload(S, R_{DO})$;
/ The following is optional */*
Delete(S, sa, mht);

Fig. 3 shows an example of the MHT constructed from the HCSI digests for the string '*aardvark*'. We will make use of Fig. 3 in running examples with the descriptions of the upcom-

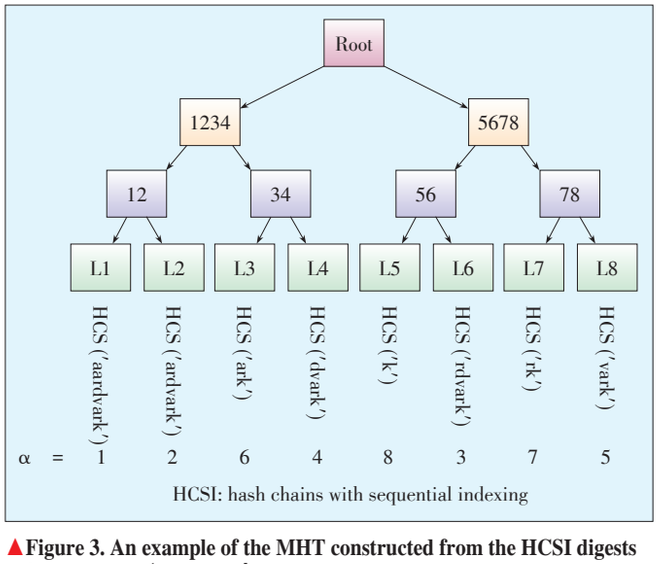
ing phases to provide some intuition as to how the scheme works.

2) Phase 2: Query

A user U submits to CS a query Q to check if substring x exists in string S. In our running example, the user submits a two different queries (to illustrate both positive and negative verification): '*%dva%*' and '*%are%*'.

3) Phase 3: Query Response

The cloud server receives a query from the user, to check for the existence of a substring in the stored string. The server constructs the HCSI digests on the suffixes of the stored string, and then proceeds to construct an MHT on the HCSI digests. This is identical to the data owner's processing in Phase 1. The server then searches for the substring in the suffix array. The result of the search returns one position from the suffix array if the substring was found, otherwise it returns two positions. If the substring was found, then the position returned is that of the matching suffix i.e. the substring is a prefix of the suffix at that position. If the substring was not found, then the two positions returned will be immediate neighbours. The first is the position of the suffix that is lexicographically smaller than the substring being sought, and the second is the position of the suffix that is lexicographically bigger. For each suffix returned, β is then calculated as shown in Definition 4.7. If the substring was found, the result is Y, otherwise it is N. The result is sent back to the client with the following verification data for the position(s) returned from the suffix array search: the position itself (i), the value of the suffix array at that position (α), the HCSI digest at position $\beta + 1$ (γ), and the verification path of the leaf for the corresponding position to the root of the MHT. In the case of a non-matching result, the first non-matching character position in the string (β), and the character at position β (S_β) is also sent back to the user. This phase is shown in **Algorithm 3**.



▲**Figure 3. An example of the MHT constructed from the HCSI digests for the string 'aardvark'.**

Algorithm 3: Query Response

Input: $Q(x)$
$sa \leftarrow BuildSA(S)$;
foreach $sfx \leftarrow \left(S_{suff(sa_S(1))} \cdots S_{suff(sa_S(|S|))}\right)$ **do**
$v_i \leftarrow HCSI(sfx)$;
end
$mht \leftarrow BuildMHT(v_1, \cdots, v_n)$;
$r \leftarrow IsSubstringOf(x, S, sa)$;
if $r.Y$ **then** /* substring was found */
$\alpha \leftarrow sa(i)$; $\gamma \leftarrow HCSI(S_{\alpha+|x|})$; $V \leftarrow \{i, \alpha, \gamma\}$;
$N \leftarrow GetSinglePathSiblings(v_i, mht)$;
else/* substring was not found */
$\alpha_l \leftarrow sa(\hat{i}_l)$; $\beta_l \leftarrow \alpha_l + |\bar{x}_l| - 1$; $\gamma_l \leftarrow HCSI(S_{\alpha_l+|\bar{x}_l|})$;
$\alpha_r \leftarrow sa(\hat{i}_r)$; $\beta_r \leftarrow \alpha_r + |\bar{x}_r| - 1$; $\gamma_r \leftarrow HCSI(S_{\alpha_r+|\bar{x}_r|})$;
$V \leftarrow \{\hat{i}_l, \alpha_l, \beta_l, S_{\beta_l}, \gamma_l, \hat{i}_r, \alpha_r, \beta_r, S_{\beta_l}, \gamma_r\}$;
$N \leftarrow GetRangePathSiblings(v_l, v_r, mht)$;
end
$VO \leftarrow BuildVO(V, N)$;
$Respond(r, VO)$;

To generate the VO for the query '*%dva%*' in our running example, the server searches for the prefix '*dva*' in the suffix array, and finds the corresponding match in L4 (Fig. 3). The server sets $\alpha = 4$, $\gamma =$ HCSI('*rk*'), and chooses the verification path for L4, corresponding to leaf L3, and internal nodes 1, 2, 5, 6, 7, and 8. α, γ. The verification path is then sent along with the response of the query (Y) to the user. To process the query '*%are%*', the server searches for the prefix '*are*' in the suffix array. The prefix is not found, so the neighbouring suffixes that are lexicographically less than and greater than '*are*' are selected, corresponding to leaves L2 and L3. The server sets $\alpha = 2$, $\beta = 4$, $S_\beta =$ 'd', and $\gamma =$ HCSI('*vark*') for L2, and $\alpha = 6$, $\beta = 8$, $S_\beta =$ '*k*', and $\gamma = null$ for L3. The server then chooses the verification path for L2 and L3, which is leaves L1 and L4, and internal node 5, 6, 7, and 8. The HCSI variables and verification path is finally sent to the user.

4) Phase 4: Verification

Upon receiving the query response from the server, the user ensures that a proof has been provided, otherwise he rejects the response. The user retrieves the latest root from the data owner, and verifies the root using the owner's public key. If the query response from the server was Y, he reconstructs the HCSI for the leaf at position i in the MHT using the reconstruction algorithm (Algorithm 1). He then uses the reconstructed leaf in conjunction with the verification path (sent by the server), to generate the MHT root. He compares the generated root with the root from the data owner. If the two roots match, he accepts the results as being correct, otherwise the result is rejected. If the query response was N, the user firstly checks that the

positions of the two suffixes returned by the server are neighbouring i.e. the left suffix position is one less than the right suffix position. This is to ensure that the server has not returned two suffixes which have suffixes in-between them, one or more of which may be suffixes that are matches for the substring. If the two positions are not neighbouring, the result is rejected. The user then constructs $\bar{x}$ for each position, by using the first $\beta-\alpha$ characters from the query substring, and appending S_β to it. Doing this allows him to reconstruct the partial suffixes for each position to the character that is the first non-matching character when compared to the query substring. He confirms that each is lexicographically smaller and larger than the query substring. If this is not the case, the result is rejected. This allows the user to ensure that the server has not simply returned two arbitrary neighbouring positions that do not in fact lexicographically border the query substring, thereby not allowing the user to confirm that the position at which the substring may be found but in fact doesn't exist. He then generates the HCSI digests for each suffix position using i, α, β, S_β, and γ (Algorithm 1). He uses the reconstructed leaf for each suffix position in conjunction with the verification path (sent by the server), to generate the MHT root. He then compares the generated root with the root from the data owner. If the two roots match, he accepts the results as being correct, otherwise the result is rejected. The verification process is outlined in **Algorithm 4**.

Algorithm 4: Query verification

Input: QR, VO
Retrieve R_{DO} from DO;
if *verify*(pk_{DO}, R_{DO}) == *false* **then** /* R_{DO} verification failed */
    reject;
end
if QR.*Y* **then** /* substring found */
    $h \leftarrow$ *ReconstructHCSI*(VO);
    $R \leftarrow$ *GenerateMHTRoot*(VO,h);
else/* substring not found */
    $h_l \leftarrow$ *ReconstructHCSI*(VO);
    $h_r \leftarrow$ *ReconstructHCSI*(VO);
    $\bar{x}_l \leftarrow x_{0\cdots\beta_l-\alpha_l} // S_{\beta_l}$; $\bar{x}_r \leftarrow x_{0\cdots\beta_r-\alpha_r} // S_{\beta_r}$;
    if $(\bar{x}_l \geqslant x) \mid (\bar{x}_r \leqslant x)$ **then**
        reject;
    end
    $R \leftarrow$ *GenerateMHTRoot*(VO, h_l,h_r);
end
if $R \neq R_{DO}^{-1}$ **then**
    reject;
end

In our running example, U receives the response Y to the query '*%dva%*'. To verify the correctness of the response, he processes the VO, also from CS, as follows: he regenerates L4 using the query '*dva*', α and γ, by calculating h(α ‖ '*d*' ‖ h($\alpha+1$ ‖ '*v*' ‖ h($\alpha+2$ ‖ '*a*' ‖ γ))) = HCSI('*dvark*'). He then regenerates the MHT root using the verification path provided by CS, and the checks against the signed root are provided by DO to ensure the generated root is correct. To verify the correctness of the second query, U runs through a similar process, but in this case, he additionally uses S_β to regenerate leaves L2 and L3. This is because the query literal '*are*' only partially matches the suffixes corresponding to leaves L2 and L4, and S_β for each suffix allows U to correctly construct each individual leaf using both part of the query and S_β. After reconstructing the leaves, he uses the verification path to reconstruct the root and checks against DO as a final step.

5) Phase 5: Update

In order to facilitate updates, the data owner simply executes the setup phase with a newer version of string S. This would generate a new MHT root that would then be used by the user to verify queries on the new string S.

5 Asymptotic Performance Analysis

We provide a brief outline of the space and time complexities achievable for both our scheme.

Table 4 shows a comparison of the complexities for the DAG scheme proposed in [1] and our scheme. The DO pre-processing phase in our scheme incurs a quarter of the storage cost of that incurred by the DAG scheme. This is not surprising as the underlying suffix arrays used in our scheme has a similar advantage over suffix trees in general. Although this advantage is in the constant factor, it in fact is a considerable advantage, and can mean the difference between a practically feasible or non-feasible solution. A similar advantage is incurred in the server query response phase, due to the fact that the server goes through a similar proof construction phase initially as that performed by the DO pre-processing phase. The VO size is small for the DAG scheme, but it increases to n as m approaches n. However, with our scheme, the size is always constantly relative to log n regardless of the size of m. This means

▼**Table 4. Space and time complexities for our scheme**

Measure	Reference [1]	Our scheme
Detect false +ve	Y	Y
Detect false -ve	N	Y
Technique used	DAG, compacted suffix tree and Hashing	MHT and suffix arrays
DO preproc.	$O(20n)+O(2n-1)H$	$O(5n)+O(2n-1)H$
Server qry resp.	$O(20n)+O(3m)+O(2n-1)H$	$O(5n+(m+2)\log n)+O(2n-1)H$
User verification	$O(m+k)H$	$O(\log n+m)H$
VO size	$O(m+k)$	$O(3+2\log n)$

In string searching theory, research papers provide asymptotic constants due to their impact on practical algorithms, and thus we also include them to allow greater precision for others when comparing our results to other work.
MHT: Merkle Hash Tree DAG: Directed Acyclic Graph

that the size of the VO under large *m* is smaller in our scheme by a log factor.

6 Empirical Evaluation

In this section, we evaluate the experiments conducted on a prototype of our proposed scheme in this section.

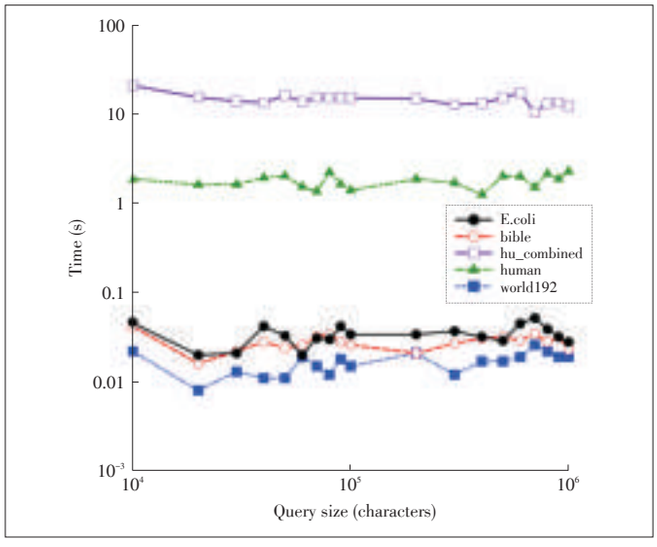
6.1 Experiment Setup

1) Client configuration: The client module was hosted and executed on a Toshiba Satellite Ultrabook U920t running Linux Ubuntu 14.04 LTS 64-bit, with 3.8 GiB RAM, 247.8 GB SSD and Intel® Core™i5-3337U CPU @ 1.80GHz x 4 processor.
2) Server configuration: The server module was run on a VMware 30 vCPU 64GB RAM CentOS 6 Linux virtual machine, which was hosted on a cluster of 19 physical servers.
3) Experiment parameters: RSA was used as the owner's signature mechanism, and the secret and public keys were generated with 2048-bits as the security parameter. The same construction was used to generate the server's secret and public keys. SHA256 was used to generate hashes for the MHT, with the digest truncated to 160 bits.
4) Prototype implementation: The prototype was implemented in C++ on both the client and server machines. Coding was initially performed on a Windows 8.1 machine with Visual Studio 2010, and was then ported to the client running on Ubuntu 14.04 with CodeBlocks 13.12 and GNU C++ 4.8.2. The suffix array construction algorithm was sourced from *libdivsufsort* that has been shown to be very efficient compared to other implementations [31]. Cryptographic functions for hashing and signatures were sourced from OpenSSL 1.0.1g. The owner-generated SA and MHT were made persistent and stored to disk (rather than temporarily creating and destroying them in memory) and 'uploaded' to the server along with the data file to facilitate query processing. From an experimental point-of-view, this facilitated ease-of-use with respect to avoiding running the same process again on the server. In practice, the server would probably re-generate both the SA and MHT independently, however this is not a requirement for the scheme to work securely. Either option (i.e. uploading the SA and MHT to the server, or independently re-generating them at the server) may be taken in practice. Consequently, the entire MHT is not loaded into memory (due to its size) by the server when processing queries. Rather, the appropriate nodes in the MHT are loaded as and when needed by the server during the VO generation phase. This serves two purposes: 1) to avoid using up large amounts of memory that could otherwise be used by other processes on the server, and 2) to reduce the overhead in loading the entire MHT into memory when queries are being processed. VOs are essentially realised as text files with an XML-like structure, without the end tags. This allows the client to recognise and parse the data in the VO in a straightforward manner.
5) Datasets: The datasets comprise of a total of five files. Three of the files were taken from the Large Canterbury Corpus [32], and two were sourced from the NCBI [33]:
   - E.coli: Complete genome of the E. Coli bacterium, size 4,638,690 b
   - bible: The King James version of the bible, size 4,047,392 b
   - world192: The CIA world fact book, size 2,473,400 b
   - human: Chromosome 10 from human genome data, size 128,985,118 b
   - hu_combined: A concatenation of chromosomes 1, 3, 5, 6, 9, and 11 from human genome data to form an approximately 1 GB file, size 1,000,003,018 b.
6) Query workload: Contiguous fragments of size 10,000 to 100,000 characters in 10,000 character increments, and from 100,000 to 1,000,000 characters in 100,000 character increments were taken from each dataset at randomly selected positions. This produced 19 query strings ranging in sizes from 10,000 characters to 1,000,000 characters for each dataset. The 19 query strings were then submitted to the server in ascending size order from the smallest query (10,000 characters) to the largest query (1,000,000 characters). The queries were processed by the server synchronously, with the query result, VO generation, and query verification for each query being performed prior to submission of the subsequent query. This series of 19 queries per dataset was repeated for a total of 30 runs per dataset to get 30 results for each individual query.

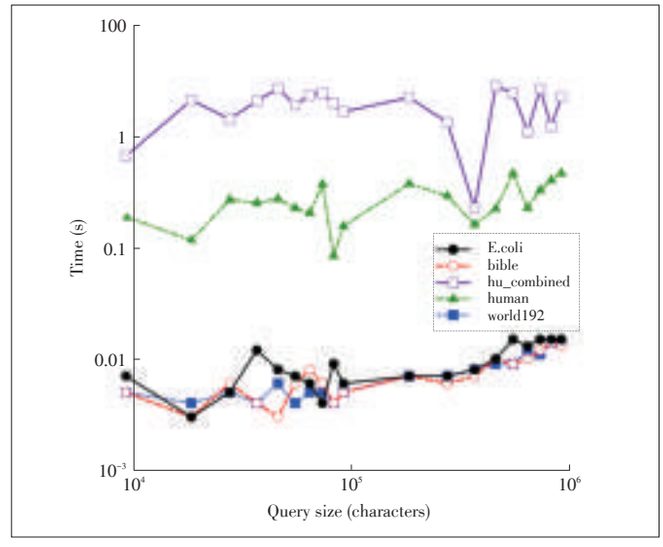
The query execution time, VO generation time, VO size and verification times were measured for each query and recorded. Averages of each of these recordings were taken for each query/dataset combinations to produce the final results as shown in the upcoming graphs.

6.2 Query Execution Times

Our research focuses on the verification of substring queries, and not the querying itself. However, we have included the query execution times as part of the results to provide a more holistic view of the implementation of the scheme. Due to the fact that the data file is not loaded into memory prior to query execution, the execution times are affected by the hits and misses due to caching. For this reason, we find that the resulting graph, in **Fig. 4**, produces slightly varying times. The size of the data file determines the difference in finding queries between files of different sizes, and so we note that the data file queries to bible, world192, and E.coli perform better on the whole than the queries to human and hu_combined. In particular, queries to the hu_combined data file takes more than 10 seconds to execute due to its comparatively larger size (1 GB) than the other data files. We also note that regardless of the query size, the query execution times remain relatively similar for queries executed on individual data files. This is ex-



▲Figure 4. Query execution times for the SA-MHT prototype.



▲Figure 5. VO generation time for the SA-MHT prototype.

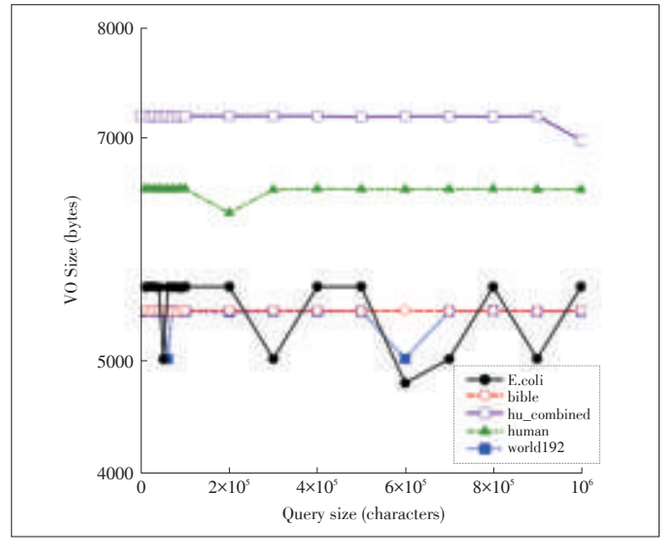
plained by the fact that a binary search is performed for each query, and occupies log $n + m$ time for each query, resulting in fairly similar times regardless of the query size.

6.3 VO Generation Time

The VO generation time is the time the SA-MHT prototype takes for the server to generate VOs for any given query. This time is incurred in addition to the query execution time and from the server's perspective. It is the cost of participating in the verification scheme per query. The results, shown in **Fig. 5**, indicate a range from 3 ms for the shorter queries to 15 ms for longer queries for the bible, world192, and E.coli data files. The human data file queries show a range VO generation times from 85 ms to 469 ms, whilst the hucombined data file shows a range from 230 ms to 2.864 s. The generation time of all VOs tends to move towards the respective upper bounds of their individual ranges as the query sizes increase. The outliers at 10,000 characters for the human data file, and 10,300 characters for the hu_combined data file could be due to cache hits as well. The VO generation phase reads the verification path nodes of the MHT from disk, node-by-node, and as such is also affected by the cache.

6.4 VO Sizes

The VO sizes seem to be bounded to a fairly constant range for each of the data files, as shown in **Fig. 6**. The VO sizes for the bible, world192, and E.coli data files seem to share a similar range of values between 4.8 KB to 5.6 KB, and this is due to the possible number of nodes in the verification path for each suffix, which is bounded by $O(\log n)$. It is worth noting that the log n bound is reflected by the jump from the lower three data files, which have almost the same log n bounds, to the human data file, which has a higher log n bound (ranging between 6.3 KB and 6.5 KB), and then another jump to the

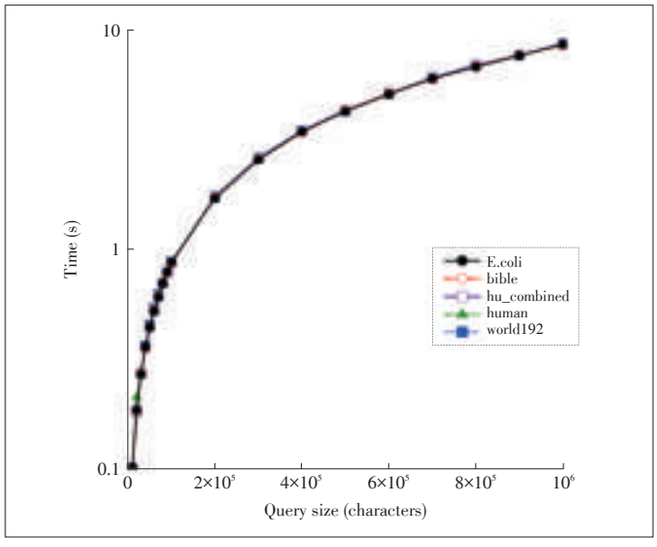


▲Figure 6. VO sizes for the SA-MHT prototype.

hu_combined data file VO sizes which has an even higher log n bound (ranging between 7 KB and 7.2 KB). The deviations from the otherwise constant values are due to fewer verification path nodes being generated for MHT nodes that happen to lie at the end of a level without a sibling (i.e. it is the end node of a level that has an odd number of nodes). In such a case, the node is not included in the verification path, and is simply promoted to the previous levels until a sibling is found. This results in fewer verification path nodes for that verification path compared to the verification path of nodes that have siblings.

6.5 Query Verification Time

Fig. 7 shows the query verification time incurred by the client. The initial observation is that regardless of data file size, the verification time for different query sizes are virtually the



▲**Figure 7. Query verification time for the SA-MHT prototype.**

same. This is a reflection of the relatively constant sizes of the VO and the fact that the difference amongst the sizes of the VOs for different data files is less than 2 KB (Fig. 6). The query size is the determining factor and this can be seen through the rise of the curve as the query sizes increase. This is a result of the number of hashes performed by the client, the maximum of which is the size of the query per suffix being verified.

6.6 Discussion on Experiment Results

The experiments on the prototype have shown promising results on the whole. The additional time spent by the server in generating the VO is largely a fraction of the query execution time, and in practice would be unnoticeable by the client. Additionally, the size of the VO is also fairly constant and does not appear to be affected much by the size of the query. The data file size causes the VO to increase, but only by a couple of kilobytes for a data file increase from 4 MB to 1 GB. Finally, the client-side verification incurs less than a second of processing time for query sizes of up to 100,000 characters, which is a large query for most applications. Larger query sizes incur more times, and are a function of the size of the query, but may still be considered usable in practice.

7 Conclusions and Future Work

We have presented an existential substring query verification scheme that meets the properties of authenticity, completeness and freshness. The scheme allows consumers to query for the existence of arbitrary substrings that are not restricted to keyword searches only, and provides verification objects with the results as proofs of correctness. Our scheme is based on suffix arrays, and provides improvements in the space and processing time in comparison to the only other comparable scheme proposed in [1]. Our scheme also provides consistently smaller VOs for large substring matches compared to the scheme proposed in [1]. The experiment results on a fully functioning prototype are promising for the applicability of our scheme to appropriate applications on the cloud.

References

[1] C. Martel, G. Nuckolls, P. Devanbu, et al., "A general model for authenticated data structures," *Algorithmica*, vol. 39, no. 1, pp. 21–41, Jan. 2004. doi: 10.1007/s00453-003-1076-8.
[2] Wikipedia. (2015 March). *Agglutination* [Online]. Available: http://en.wikipedia.org/w/index.php?title=Agglutination&oldid=648155093
[3] G. Ateniese, R. Burns, R. Curtmola, et al., "Provable data possession at untrusted stores," in *Proc. 14th ACM Conference on Computer and Communications Security*, Alexandria, USA, 2007, pp. 598–609. doi: 10.1145/1315245.1315318.
[4] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proc. 4th International Conference on Security and Privacy in Communication Networks*, Istanbul, Turkey, Article 9, 2008, pp. 9:1–9:10. doi: 10.1145/1460877.1460889.
[5] A. Juels and B. S. Kaliski, "PORS: proofs of retrievability for large files," in *Proc. 14th ACM Conference on Computer and Communications Security*, Alexandria, USA, 2007, pp. 584–597. doi: 10.1145/1315245.1315317.
[6] T. S. J. Schwarz and E. L. Miller, "Store, forget, and check: using algebraic signatures to check remotely administered storage," in *26th IEEE International Conference on Distributed Computing Systems*, Lisboa, Portugal, 2006, pp. 12–12. doi: 10.1109/ICDCS.2006.80.
[7] F. Li, M. Hadjieleftheriou, G. Kollios, and L. Reyzin, "Dynamic authenticated index structures for outsourced databases," in *Proc. 2006 ACM SIGMOD International Conference on Management of Data*, Chicago, USA, 2006, pp. 121–132. doi: 10.1145/1142473.1142488.
[8] E. Mykletun, M. Narasimha, and G. Tsudik, "Providing authentication and integrity in outsourced databases using merkle hash trees," UCI-SCONCE Technical Report, 2003.
[9] M. Narasimha and G. Tsudik, "DSAC: integrity for outsourced databases with signature aggregation and chaining," in *Proc. 14th ACM International Conference on Information and Knowledge Management*, Bremen, Germany, 2005, pp. 235–236. doi: 10.1145/1099554.1099604.
[10] H. Pang and K.-L. Tan, "Verifying completeness of relational query answers from online servers," *ACM Transactions on Information and System Security*, vol. 11, no. 2, Article 5, Mar. 2008. doi: 10.1145/1330332.1330337.
[11] Q. Zheng, S. Xu, and G. Ateniese, "Efficient query integrity for outsourced dynamic databases," in *Proc. 2012 ACM Workshop on Cloud Computing Security Workshop*, Raleigh, USA, 2012, pp. 71–82. doi: 10.1145/2381913.2381927.
[12] K. Mouratidis, D. Sacharidis, and H. Pang, "Partially materialized digest scheme: an efficient verification method for outsourced databases," *The VLDB Journal*, vol. 18, no. 1, pp. 363–381, Jan. 2009. doi: 10.1007/s00778-008-0108-z.
[13] S. Singh and S. Prabhakar, "Ensuring correctness over untrusted private database," in *Proc. 11th International Conference on Extending Database Technology: Advances in Database Technology*, Nantes, France, 2008, pp. 476–486. doi: 10.1145/1353343.1353402.
[14] M. T. Goodrich, R. Tamassia, and N. Triandopoulos, "Super-efficient verification of dynamic outsourced databases," in *Proc. The Cryptopgraphers' Track at the RSA conference on Topics in Cryptology (CT-RSA'08)*, San Francisco, USA, Apr. 2008, pp. 407–424. doi: 10.1007/978-3-540-79263-5_26.
[15] M. Noferesti, M. A. Hadavi, and R. Jalili, "A signature-based approach of correctness assurance in data outsourcing scenarios," *Information Systems Security*, vol. 7093, S. Jajodia and C. Mazumdar, Eds. Germany: Springer Berlin Heidelberg, 2011, pp. 374–378. doi: 10.1007/978-3-642-25560-1_26.
[16] T. K. Dang, "Ensuring correctness, completeness, and freshness for outsourced tree-indexed data," *Information Resources Management Journal*, vol. 21, no. 1, pp. 59–76, Jan. 2008. doi: 10.4018/irmj.2008010104.
[17] M. Narasimha and G. Tsudik, "Authentication of outsourced databases using signature aggregation and chaining," in *Proc. 11th International Conference on Database Systems for Advanced Applications (DASFAA'06)*, Singapore, Apr. 2006, pp. 420–436. doi: 10.1007/1173383630.
[18] M. Xie, H. Wang, J. Yin, and X. Meng, "Providing freshness guarantees for out-

sourced databases," in *Proc. 11th International Conference on Extending Database Technology: Advances in Database Technology (EDBT '08)*, New York, USA, 2008, pp. 323–332. doi: 10.1145/1353343.1353384.
[19] R. Jain and S. Prabhakar, "Trustworthy data from untrusted databases," in *IEEE 29th International Conference on Data Engineering (ICDE)*, Brisbane, Australia, Apr. 2013, pp. 529–540. doi: 10.1109/ICDE.2013.6544853.
[20] Y. Zhou and C. Wang, "A query verification method for making outsourced databases trustworthy," in *IEEE Ninth International Conference on Services Computing (SCC)*, Honolulu, USA, Jun. 2012, pp. 298– 305. doi: 10.1109/SCC.2012.63.
[21] G. Nuckolls, "Verified query results from hybrid authentication trees," in *Data and Applications Security XIX*, vol. 3654, S. Jajodia and D. Wijesekera, Eds. Springer Berlin Heidelberg, 2005, pp. 84–98. doi: 10.1007/11535706_7.
[22] B. Palazzi, M. Pizzonia, and S. Pucacco, "Query Racing: Fast Completeness Certification of Query Results," in *Data and Applications Security and Privacy XXIV*, vol. 6166, S. Foresti and S. Jajodia, Eds. Germany: Springer Berlin Heidelberg, 2010, pp. 177–192. doi: 10.1007/978-3-642-13739-6_12.
[23] H. Pang and K. Mouratidis. "Authenticating the query results of text search engines," *Proc. VLDB Endow.*, vol. 1, no. 1, pp. 126 - 137, Aug. 2008. doi: 10.14778/1453856.1453875.
[24] M. T. Goodrich, C. Papamanthou, D. Nguyen, et al., "Efficient verification of web-content searching through authenticated web crawlers," *Proc. VLDB Endow.*, vol. 5, no. 10, pp. 920-931, Jun. 2012. doi: 10.14778/2336664.2336666
[25] R. Rivest, "The MD5 message-digest algorithm," IETF RFC1321, 1992.
[26] *Secure Hash Standard, Federal Information Processing Standard (FIPS)*, FIPS 180-2, Aug. 2002.
[27] *Secure Hash Standard, Federal Information Processing Standard (FIPS)*, FIPS 180-4, Mar. 2012.
[28] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978. doi: 10.1145/359340.359342.
[29] R. C. Merkle, "Protocols for public key cryptosystems," in *IEEE Symposium on Security and Privacy*, Oakland, USA, Apr. 1980, pp. 122–134. doi: 10.1109/SP.1980.10006.
[30] M. T. Goodrich, R. Tamassia, and A. Schwerin, "Implementation of an authenticated dictionary with skip lists and commutative hashing," in *DARPA Information Survivability Conference & Exposition II*, Anaheim, USA, Jun. 2001. pp. 68–82. doi: 10.1109/DISCEX.2001.932160.
[31] Y. Mori. (2014). *The Benchmark Results of Implementations of Various, Latest Suffix Array Construction Algorithms* [online]. Available: https://code.google.com/p/libdivsufsort/wiki/SACABenchmarks
[32] T. Bell. (2014). *The Large Canterbury Corpus* [online]. Available: http://corpus.canterbury.ac.nz/descriptions/#large
[33] National Center for Biotechnology Information. (2014). *Homo-Sapien Genome* [online]. Available: ftp://ftp.ncbi.nlm.nih.gov/genomes/H_sapiens/

Manuscript received: 2016-04-09

Biographies

Faizal Riaz-ud-Din (faizal.din@ieee.org) has a background in database and software development. He has been working in academia as well as in industry for a number of years and is currently pursuing a doctorate part-time. His interests lie in the area of query verification in databases and text-based data on the cloud.

Robin Doss (robin.doss@deakin.edu.au) received the BEng from the University of Madras, India, in 1999, and the MEng and PhD degrees from the Royal Melbourne Institute of Technology (RMIT), Australia, in 2000 and 2004, respectively. He has held professional appointments with Ericsson Australia, RMIT University, and IBM Research, Switzerland. He joined Deakin University, Australia, in 2003, and is currently a senior lecturer in computing. Since 2003, he has published more than 50 papers in refereed international journals, international conference proceedings and technical reports for industry and government. His current research interests are in the broad areas of communication systems, protocol design, wireless networks, security and privacy. He is a member of the IEEE.

Roundup

Introduction to *ZTE Communications*

ZTE Communications is a quarterly, peer-reviewed international technical journal (ISSN 1673–5188 and CODEN ZCTOAK) sponsored by ZTE Corporation, a major international provider of telecommunications, enterprise and consumer technology solutions for the Mobile Internet. The journal publishes original academic papers and research findings on the whole range of communications topics, including communications and information system design, optical fiber and electro-optical engineering, microwave technology, radio wave propagation, antenna engineering, electromagnetics, signal and image processing, and power engineering. The journal is designed to be an integrated forum for university academics and industry researchers from around the world. *ZTE Communications* was founded in 2003 and has a readership of 5500. The English version is distributed to universities, colleges, and research institutes in more than 140 countries. It is listed in Inspec, Cambridge Scientific Abstracts (CSA), Index of Copernicus (IC), Ulrich's Periodicals Directory, Norwegian Social Science Data Services (NSD), Chinese Journal Fulltext Databases, Wanfang Data — Digital Periodicals, and China Science and Technology Journal Database. Each issue of *ZTE Communications* is based around a Special Topic, and past issues have attracted contributions from leading international experts in their fields.

DOI: 10.3969/j. issn. 1673-5188. 2016. S0. 004
http://www.cnki.net/kcms/detail/34.1294.TN.20160613.1553.002.html, published online June 13, 2016

A Secure Key Management Scheme for Heterogeneous Secure Vehicular Communication Systems

LEI Ao, Chibueze Ogah, Philip Asuquo, Haitham Cruickshank, and SUN Zhili
(Institute for Communication Systems (ICS), University of Surrey, Guildford, GU2 7XH, United Kingdom)

Abstract

Intelligent transportation system (ITS) is proposed as the most effective way to improve road safety and traffic efficiency. However, the future of ITS for large scale transportation infrastructures deployment highly depends on the security level of vehicular communication systems (VCS). Security applications in VCS are fulfilled through secured group broadcast. Therefore, secure key management schemes are considered as a critical research topic for network security. In this paper, we propose a framework for providing secure key management within heterogeneous network. The security managers (SMs) play a key role in the framework by retrieving the vehicle departure information, encapsulating block to transport keys and then executing rekeying to vehicles within the same security domain. The first part of this framework is a novel Group Key Management (GKM) scheme basing on leaving probability (LP) of vehicles to depart current VCS region. Vehicle´s LP factor is introduced into GKM scheme to achieve a more efficient rekeying scheme and less rekeying costs. The second component of the framework using the blockchain concept to simplify the distributed key management in heterogeneous VCS domains. Extensive simulations and analysis are provided to show the effectiveness and efficiency of the proposed framework: Our GKM results demonstrate that probability-based BR reduces rekeying cost compared to the benchmark scheme, while the blockchain decreases the time cost of key transmission over heterogeneous networks.

Keywords

leaving probability; blockchain; group key management; heterogeneous; vehicular communication systems (VCS)

1 Introduction

Vehicular communication systems (VCS) supports not only message exchange among vehicles, but between cars and infrastructure facilities as well. Infrastructure access points in VCS are called Road Side Units (RSUs) [1]. RSU acts as a base station in VCS and covers a small section on the road. Traditional VCS is comprised of multiple RSU cells and offers a platform among intelligent transportation systems (ITS) for vehicles to exchange different kinds of messages such as safety notification messages. With the help of VCS, ITS can offer a more safe and efficient traffic management, which is the basic function of ITS. Moreover, commercial applications, such as electric vehicle charging [2], can be implemented on a dedicated platform. A recent report from U.S Department of Transport (DoT) shows that 82% of the accidents can be prevented by using ITS systems [3]. Even though significant developments have taken place over the past few years in the area of VCS, security issues, especially key management schemes are still an important topic for research. High mobility, large volume, frequent handoff of vehicular nodes and heterogeneity networks pose different challenges compared to the traditional mobile networks.

ITS spans across a wide range of applications which are classified into two categories: vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) [4]. VCS security highly relies on the safety for exchange of beacon messages. These beacon messages are usually referred to as Cooperative Awareness Messages (CAMs) for EU [5] or Basic Safety Messages (BSMs) for US [6], as they enable other vehicles to be aware of their surroundings. Vehicles located in the same RSU cell form a group and the current traffic situation is generated based on the summary of BSMs broadcasted by other group members. The trustfulness and legality of BSM information is proved by encrypting safety messages with a pre-agreed group key (GK). For this reason, the problem of providing ITS security can be mapped into the problem of how to reliably distribute or update group keys among all the communicating participants. Several group key management (GKM) approaches for mobile networks (e.g Logical Key Hierarchy (LKH) and One-way Function Tree) have been presented in recent years. Unfortunately, these approaches are quite inefficient for VCS application due to huge number and high variability in vehicular nodes. Hence, there is need for a novel and more efficient key management scheme

for VCS.

To meet the security requirements, GK has to be refreshed and redistributed (rekeying) securely whenever a group member changes in order to achieve forward and backward secrecy [7]. This approach poses the challenge of rekeying efficiency. Several approaches aim to improve efficiency of managing keys for group nodes, and schemes for individual node rekeying like key tree approaches [8], [9] are developed to ease the problem. Furthermore, Batch Rekeying (BR) [10] is proposed to significantly improve efficiency compared to individual rekeying schemes. But these approaches are not suitable for VCS application as the number of mobility nodes may be huge in VCS. The authors in [11] introduce BR into multiple key trees and select the tree with less rekeying cost upon rekeying. However, nodes in [11] are traditional mobile nodes with irregular trajectory. Paper [12] presents a GKM scheme for Internet of Things applications, including VCS scenarios. Based on the idea in [10], authors introduce their method for VCS but mainly focus on key initialisation and registration stage.

Aside from the aforementioned problem, heterogeneity issues are an inevitable aspect in wireless networks. Heterogeneity in wireless network refers to either the difference on the traffic volumes, or distinct network structures. Heterogeneous traffic volumes are classified as nodes densities or message traffic capabilities [13] while the heterogeneous network structures normally stand for the network managed under different topologies [14], [15] or central managers. These heterogeneities are the major considerations in evaluating the essential requirements of VCS key management scheme. Recently, heterogeneous vehicular communication networks are given more attention. The heterogeneity in terms of different central authorities has become a reality problem as VCS is considered as a worldwide system covering multiple countries. Specifically speaking, two RSUs in different security domains should be able to keep understanding messages from the same car passing through their common border between domains. With this in mind, user cross-domain hand-offs must not be overlooked in VCS.

In this paper, we propose a key management scheme for VCS scenarios, including the group batch rekeying scheme and key transmission between two heterogeneous networks. Different form the previous group batch rekeying schemes [7], [10]–[12], Leaving probability (LP) is introduced into the proposed scheme to further reduce rekeying cost in order to achieve better efficiency. Furthermore, with the help of blockchain, a simplified handshake procedure is achieved for heterogeneous networks. Performance evaluations of this paper demonstrate that LP approach achieves much less rekeying overhead compared to the benchmark BR scheme. The time consumption result of heterogeneous key management approach is compared with that in traditional network structure to prove that the blockchain concept helps to shorten the key transmission handshake time.

The remainder of this paper are organised as follows: Section 2 briefly introduces key management techniques. Model overview and details of our scheme are displayed in section 3. We describe our system model, and then introduce the rest part of our ideas, namely, LP, vehicle initialisation procedures and key transmission between heterogeneous networks. Scenario parameter assumptions, key registration procedures, rekeying costs and blockchain performance are analysed in Section 4. Section 5 concludes the paper and presents some future plans.

2 Related Work

2.1 Key Tree Approach

Key tree approaches include the key graph approach and LKH, which are scalable structures to manage large volume of nodes. Hierarchy tree reduces the processing complexity of each member change request from $O(N)$ to $O(\log_d N)$ [16], where d is the degree of key tree and N is the group size. In key tree structure, GK is placed at the root of the tree. It is called to encrypt messages whenever a member wants to exchange messages with others. This means that all the group members own a copy of GK and these members knows all the details about the GK, so the GK must be a symmetric encryption scheme key, such as Advanced Encryption Standard (AES) [17]. Individual keys (IK) are located at leaf nodes of the tree, they are user nodes in the broadcasting group. The rest of tree nodes are logical key nodes which are used to encrypt parent keys, called Key Encryption Key (KEK) [7]. In wireless network, mobility nodes form the mainstream composition of network, especially in VCS. Therefore BR is a critical method to reduce a large proportion of rekeying messages, which is caused by individual rekeying. To eliminate high rekeying cost in individual rekeying, BR scheme collects all member modification requests within a certain period of time and triggers rekeying broadcasting at end of the period. In this way, key manager aggregates multiple broadcast messages into a single one and achieves much better efficiency, where the period of time is batch interval t_{BR} and end of the period is called batch edge.

GKM algorithm in [10] is the first scheme using batch conception and it was cited by large number of batch rekeying papers. The authors assume there are J vehicles joining the group and L vehicles leaving, respectively. Four situations are classified as follows:

- Case-1: If $J = L$, new joining users replace the previous places of leaving users.
- Case-2: If $J < L$, joining members fill into J minimum-depth tree leaf vacancies of the departing users.
- Case-3: If $J > L$ and $L = 0$, the key manager first finds the shallowest node and remove it, then forms the node and joining users as a new subtree. Finally the key manager inserts

the tree at the deleted point.

- Case-4: If $J > L$ and $L > 0$, the central manager executes steps in case-2 first and operates algorithm in case-3 afterwards.

It is a framework for all mobile networks, but not dedicated for VCS applications. With this in mind, the probability factor in VCS scenarios can be involved in joining member ordering and inserting point selection as well. Details are discussed in section 3.

2.2 Blockchain Applications

A lot of attention has been attracted to the blockchain concept since its parent production, bitcoin, was launched in late 2008 [18]. The core idea of blockchain is that it maintains a distributed and synchronised ledger of transactions. It benefits to accountability function by using block look-up, which is fairly useful since the malicious user must be revoked in time. More importantly, a transaction can be used to transmit information among decentralised network. Even through there is no centralised manager, the key to maintain the information correctness and integrity in blockchain network is that all the blocks are distributed verified by large of network participants (miners) [19].

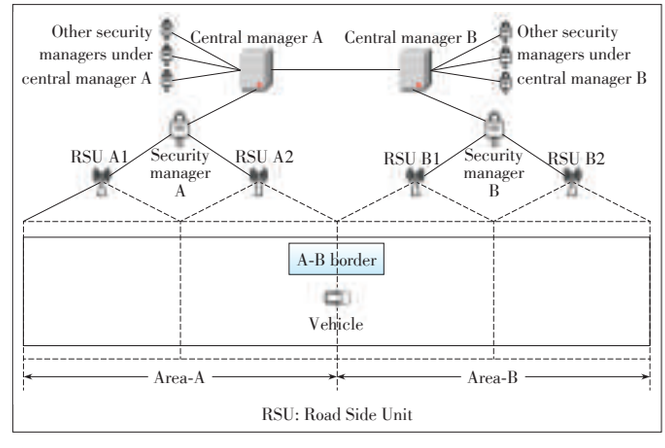
To the best of our knowledge, no previous works have adopted the blockchain mechanism to transmit information for wireless network applications, let alone the VCS applications. In this paper, we utilise the Security Managers (SM) network to transmit and verify vehicle keys in the across border requests, rather than forwarding them to the third party authorities.

3 Proposed Framework

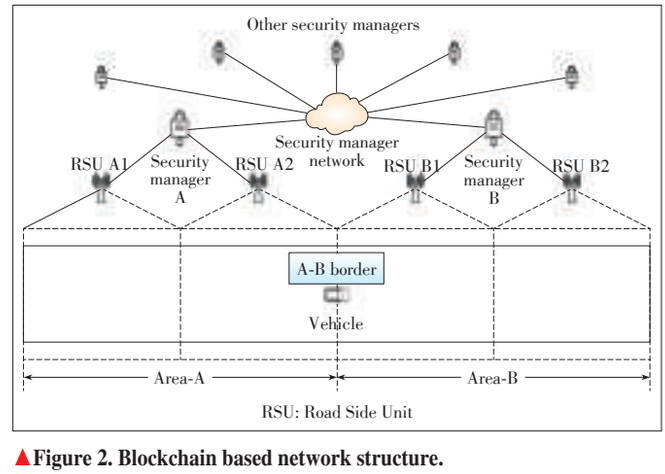
3.1 System Model

We focus exclusively on a system of vehicles each equipping an On Board Unit (OBU) embedded with wireless communication module based on the IEEE 802.11p standard. The OBU enables vehicles to communicate with nearby vehicles and infrastructures (RSU). The RSUs are equipped with the identical wireless standard. Security managers are placed on the upper level of RSUs and their logical coverage area is called security domain. As demonstrated in **Fig. 1**, Area-A is a security domain which is managed by security manager A. This traditional network structure employs central managers (or trusted third party authority) at top of the network to manage cryptography materials, this however makes it an inefficient key exchange, and will require supernumerary handshakes if a car passes from one security domain to another. The key transportation achieved by our approach could thus be simplified by using blockchain mining method, meaning the messages will be verified by SM network but not third party authorities. For instance, let us consider a scenario in which two cars in same security domain apply to depart, each on going in a different direction. The trust authority must send two distinct messages in order to finish key transmission. In our model, on the other hand, simplifies the network structure, specifically the trust third party authorities. Similar to the bitcoin network, the function of blockchain enable nodes to share information without the need for a central party to secure this ledger. The trusted third party authorities only take part in distributing initial keys, while the cryptography issues are computed by SM, which is placed at higher level of the network. As shown in **Fig. 2**, SM is connected with a "cloud" that may link with SMs on other domains and certification entities with a territory.

A key management scheme has three functional components: key initialisation, group key management and key transmission between heterogeneous networks [16], [20]. Our model assumes that the key initialisation is managed by the third party central authorities. We suppose the central authorities have secure communication link with SMs. Therefore, authorities are responsible for generating the permanent vehicle identities only. Vehicles travel on a road and periodically transmit safety messages using OBU, which are collected by RSU that are built along the road at regular intervals. The RSU forwards received messages to the upper level SM to verify the authentici-



▲Figure 1. Traditional network structure.



▲Figure 2. Blockchain based network structure.

ty of such messages. The aforementioned group key management is executed by SMs. They start their rekeying work by using wireless IEEE 802.11p broadcasting within their own security domain, which is triggered depending on member alteration. The messages are supposed to share with neighbouring SMs to transport keys if they indicate a SM-border-crossing action. Similar to bitcoin applications, the crossing border actions are encapsulated into transactions and a block is formed by multiple transactions within a short period of time. Aside from this, the SMs take the role of miners. Our proposal is to transport keys by mining blocks so that a blockchain can be maintained for heterogeneous key management purpose, at least within a local SM domain. As a result, the list of new joining members is delivered by retrieving the information from a block.

3.2 Probability Based Group Key Management

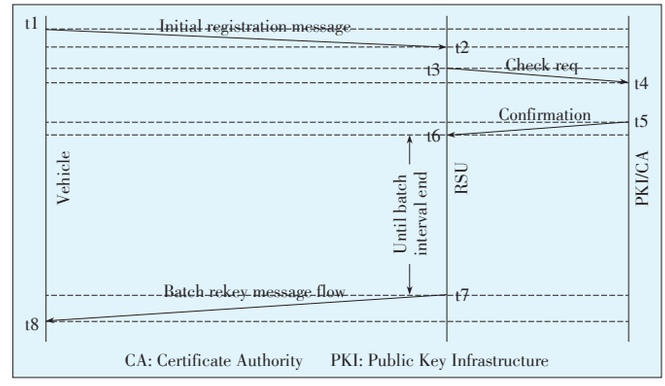
A key management scheme has three functional components: key initialisation, group key management and key transmission between heterogeneous networks [16], [20]. Our model assumes that the key initialisation is managed by the third party central authorities. We suppose the central authorities have secure communication link with SMs. Therefore, authorities are responsible for generating the permanent vehicle identities only.

3.2.1 Joining Handshake

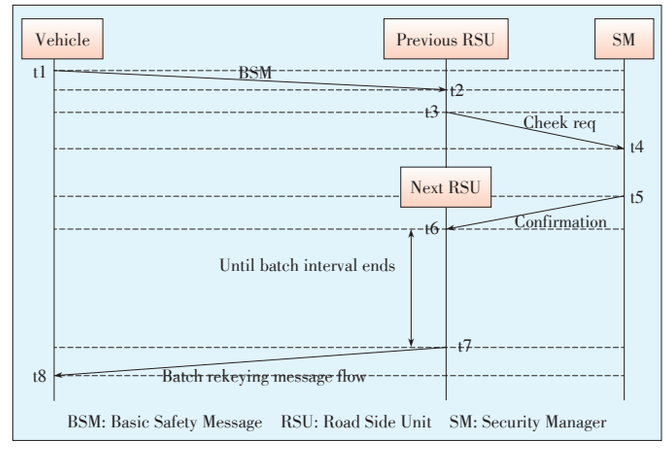
Cryptographic encryption schemes and certificates are introduced to provide security in ITS [21]. Public/private key pairs and certificates are managed by Public Key Infrastructure (PKI) and Certificate Authority (CA), respectively. IEEE1609.2 [21] defines the use of powerful cryptographic schemes such as Elliptic Curve Integrated Encryption (ECIES) [22] for individual encryption (encrypting the rekeying block only for a single user but not for a group), which requires more processing resources. AES is used for group communication, which is considered as a lightweight symmetrical encryption algorithm. In our scheme, all vehicles hold either permanent or temporary certificate in order to complete joining handshake work. A temporary certificate is assigned before vehicle leaves the manufacturer. As shown in **Fig. 3**, new vehicles need to use the temporary certificateto send an Initial Registration Message (IRM) for self-registration at initial participation in ITS environment.

Permanent certificates become effective whenever a vehicle changes to another RSU area under the same security domain. The SM checks the correctness of the safety beacon messages. In this case, a new RSU obtains the region changing information from the verified safety beacon messages. **Fig. 4** illustrates the above procedures. For the above situations, SM and RSUs need to collect vehicle entry and exit information via BSMs or IRMs to achieve batch rekeying.

When a vehicle attempts to move into a new RSU area that



▲**Figure 3. Vehicle initial joining handshake.**



▲**Figure 4. Vehicle RSU changing handshake.**

is under administration of the same SM, it keeps broadcasting BSMs using previous GK: $AES\{Info, GK\} + ECDSA\{Cip, K_{priv}\} + Cert_p$, where $Info$ is the safety information, K_{priv} is private key of vehicle and Cip is ciphertext. Permanent certificate $Cert_p$ includes authorised receipt to prove that the certificate holder possesses a legal digital receipt and public/private key pairs which are authenticated by local SM. The RSU forwards the certificate and signature to the applications layer of SM, after receiving the check request. Digital signature scheme Elliptic Curve Digital Signature Algorithm (ECDSA) [23] is used in our scenario to provide better degree of security. The legality of the vehicle's identity is verified by SM and a confirmation message is then sent back to RSU. The RSU starts to prepare the rekeying message upon "Confirmation" receipt. The rekeying broadcast is sent until the start of next batch interval. We assume that both pervious and new RSU can receive the BSM. Thus, previous RSU knows the leaving activity, while the new RSU obtaining information from the same BSM as RSUs are designed to store GK of its neighbours.

3.2.2 Leaving Probability

LP of mobile node is defined in [24] as an average number of nodes leaving the group within a rekeying interval. For tradi-

tional mobile networks (e.g. 3G, LTE and 5G network), entrance and departure of portable nodes are unpredictable. Hence some key management schemes require nodes subscribing several rekeying intervals in order to calculate leaving probability. Unfortunately, security vulnerabilities appear when system allows users to select their own subscription period: a malicious user eavesdrops critical messages by asking active period longer then its real residence time.

Probability models are much easier to implement for vehicle nodes in VCS since they have predictable moving trajectory. With this in mind, a dedicated LP calculation algorithm is needed for VCS scenarios. According to the traffic survey [25] at a one-directional urban road, speed distribution fits normal distribution function in (1) [26].

$$f(x|\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{(x-\mu)^2}{2\sigma^2}}, \tag{1}$$

where μ is the mean or expectation of the distribution and σ is the standard deviation. With the help of the speed distribution and vehicle specifications, the central manager is able to compute the possible speed range (PSR) and possible departure speed range (PDSR). The upper boundary U_{PSR} stands for the maximum speed in which vehicle can reach at end of current batch interval (t_{BR}). Similar to U_{PSR} , L_{PSR} is for the minimum speed if vehicle tries to slow down. In addition, U_{PSDR} and L_{PSDR} are the highest and lowest speed for car to leave the current RSU coverage, respectively. We assuming $d_{\mathrm{Re}main}$ is the distance between the vehicle current position and coverage border which is directly ahead of the vehicle. Thus, the leaving probability P_L is calculated as (2):

$$P_L=\frac{\int_{L_{PSDR}}^{U_{PSDR}}f(x|\mu,\sigma)\mathrm{d}x}{\int_{L_{PSR}}^{U_{PSR}}f(x|\mu,\sigma)\mathrm{d}x}. \tag{2}$$

RSU knows vehicle's maximum positive and negative acceleration by listening to the safety beacon messages, thus it is easier to calculate the upper and lower boundary ($V_{\max}$ and $V_{\min}$) of PSR. For PSDR boundaries, $V_{dep-\max}$ stands for the maximum speed for vehicle to depart. There are two different extreme situations:

1) The vehicle keeps speed-up with maximum positive acceleration a_+ until the speed reaches $V_{\max}$. The speed is kept until the end of the batch interval. The overall distance d_{Remain} is covered by the vehicle.
2) The vehicle already has enough speed and d_{Remain} is short enough so that the vehicle is able to leave the region easily. The vehicle speeds up with an acceleration lower than a_+ . It reaches $V_{\max}$ at mid of t_{BR} and keeps the speed $V_{\max}$ until the end of the batch interval.

Similarly, there are two possible situations of $V_{dep-\min}$:

1) The current speed $V_{current}$ is fast enough for vehicle to leave the RSU region, therefore the minimum speed for the vehicle to leave $V_{dep-\min}$ is decided by decreasing speed until the end of t_{BR} under the assumption that the node can travel d_{Remain} .
2) The vehicle has to speed up in order to depart in t_{BR} , therefore the node first improves the current speed from $V_{current}$ to $V_{dep-\min}$, and then keeps it until the end of the batch interval.

According to the possibilities above, the first situation is that the vehicle can leave the region only by driving with current speed:

$$V_{dep-\min}=\frac{2\cdot d_{remain}}{t_{BR}}-V_{current}. \tag{3}$$

Here it is assumed vehicle spends time t_1 speeding up to $V_{dep-\max}$, and $V_{dep-\max}$ are calculated in (4):

$$\begin{cases} V_{dep-\max}=V_{Current}+t_1\cdot a_+ \\ d_{remain}=V_{dep-\max}\cdot(t_{BR}-t_1) \\ +0.5\cdot t_1\cdot(V_{dep-\max}+V_{current}) \end{cases}. \tag{4}$$

Therefore, $V_{dep-\max}$ is computed by a summarised equation:

$$V_{dep-\min}=V_{current}+a_+\cdot t_{BR}+\sqrt{2\cdot a_+(V_{current}\cdot t_{BR}-d_{remain})+a_+^2\cdot t_{BR}^2}. \tag{5}$$

To sum up, LP can be generated by using **Algorithm 1**.

Algorithm 1 Leaving Probability Calculation

Input: Current speed $V_{Current}$, distance to coverage border d_{remain} , vehicle maximum positive and negative acceleration a_+ & a_- , batch interval t_{BR} , maximum speed the vehicle can reach V_{limit}

Output: Leaving Probability (*LP*): P_L

Max speed in t_{BR} : $V_{\max-expect}=V_{current}+a_+\cdot t_{BR}$
if ($V_{\min-expect}\geqslant V_{limit}$) **then**
    $d_{\max}$ in t_{BR} , keep gaining speed until V_{limit} ;
else
    $d_{\max}$ in t_{BR} , keep gaining speed until $V_{\max-expect}$;
endif
if ($d_{\max}\geqslant d_{remain}$) **then**
    $V_{dep-\max}=\min(V_{\max-expect}, V_{limit})$;
else
    Set *LP* for this node $P_u=0$;
endif
MIN speed in t_{BR} : $V_{min-expect}=V_{current}-t_{BR}\times a_-$;
if ($V_{current}\ 6\cdot t_{BR}\geqslant d_{remain}$) **then**
    call equation (3) to calculate $V_{dep-\min}$;
else
    call equation (5) to calculate $V_{dep-\min}$;
endif
Calculate maximum and minimum possible speed of the vehicle, $V_{\max}$ and $V_{\min}$;
LP is calculated by employing $V_{dep-\max}$, $V_{dep-\min}$, $V_{\max}$ and $V_{\min}$ into equation (2);

20: **End Algorithm**

3.2.3 Leaving Ratio

However, in a VCS scenario, most of the vehicles have no chance to leave the communication group before the next batch edge since it is impossible for them to reach the speed to leave the region border in rekeying interval. For this reason, another parameter Leaving Ratio (LR) is involved to substitute LP. Within the range (0, 1], LR is a ratio of the rekeying interval and time cost for the vehicle leaving the broadcast border. Similar to the definition of LP, LR represents the inverse of the number of rekeying intervals using for a vehicle to leave the group:

$$LR = \min\left(1, \frac{t_{BR}}{t_{out}}\right). \quad (6)$$

The parameter t_{out} is the time cost for a vehicle to leave, which is computed by (7):

$$t_{out} = \frac{d_{remain}}{V_{current}}. \quad (7)$$

3.2.4 Joining User Sequence

According to the batch rekeying scheme in [10], new joining users have two circumstances to be attached to the key tree:

1) New joining users fill into the vacancies caused by departure users.
2) New users joining the subtree form a subtree and the subtree is inserted into the key tree.

Both the circumstances are related to inserting fresh nodes in order of LP and LR values. In our scenario, nodes are arranged according to LP and LR with either positive or negative sequence. LP is considered with higher priority compared to LR during work arrangement. LR is taken into operating if the rest of the nodes are with LP equal to zero. For example, if the joining users are arranged with leaving probabilities from high to low, the sequence should be $LP_{high} > LP_{med} > LP_{low} > LR_{high} > LR_{mid} > LR_{low}$.

3.3 Heterogeneous Key Management

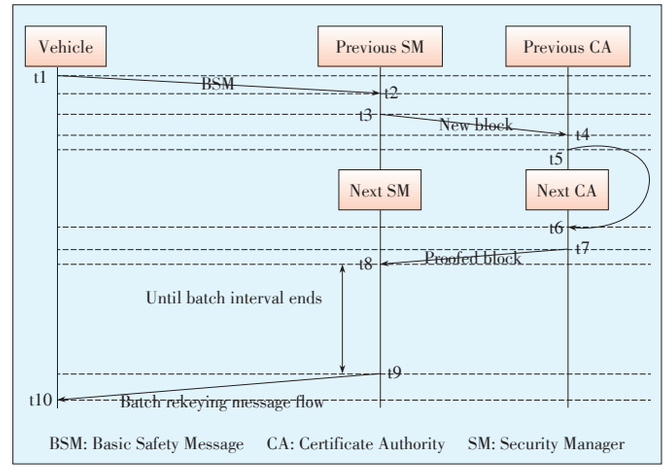
We propose the blockchain concept for heterogeneous key management, which aims to simplify the distributed key management in large heterogeneous security domains. A lightweight and scalable key transmission scheme is implemented in our scheme by using blockchain.

3.3.1 Heterogeneous Key Management

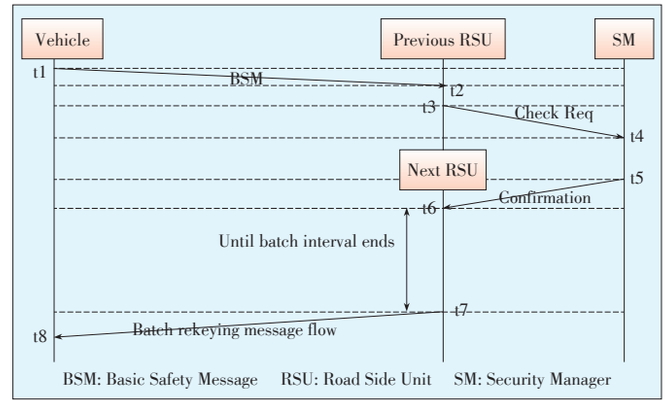
The handshake process in the traditional network is shown in **Fig. 5**. When a vehicle attempts to join a new geography territory in which infrastructures are managed by a new certificate authority (CA), the old CA picks up this border crossing activity from the beacon messages that are sent by the vehicle. Then it generates a border crossing request along with useful information related to the vehicle and forwards all these materials to the next CA. A new group key will be sent to the vehicle after new CA has verified such cryptography materials. This however delays the key transmission between two security domains. The blockchain concept is one approach to facilitate this, because it eliminates the third party authorities and allows decentralised key transmission between networks. We abstract handshake steps of blockchain network (**Fig. 6**). In our model, border crossing requests are gathered into transactions, these transactions are further collected into a candidate block. This candidate block is then distributed into the SM network for other SMs to verify, which follows the mining processes in bitcoin network [19]. The mined block is returned back to SM network after the solution of the proof-of-work has been found [27] and the destination SM retrieves the joining vehicle information from this block.

3.3.2 Transaction Format

Transactions are designed to encapsulate key transmission materials from the source SM to destination SM. Six fields are contained in the transaction of our model (**Table 1**). Hash in



▲**Figure 5. Heterogeneous key management in traditional network.**



▲**Figure 6. Heterogeneous key management in blockchain based network.**

▼Table 1. The transaction format

Field	Description
Transaction hash	A Hash of the transaction
Transaction number	Number of this transaction in block
Current security manager	Current security domain
Destination security manager	Next security domain
Vehicle identity (current pseudonym)	Current vehicle pseudonym
Vehicle certificate	Certificate of the pseudonym

the first field aims to simplify computation burden of miners. The destination SM knows the existence of new joining vehicles from the fourth field if the value in this field matches the SM identity. Even more important, the destination SM can encrypt the rekeying message by using the vehicle public key, which is embedded in vehicle certificate in the last field. As one of the most important metric to measure the performance of blockchain, the number of transactions in bitcoin related research is how many transactions are mined in a second. However in VCS scenarios, we use an alternative definition, which is the average number of transactions in a block.

3.3.3 Block Format

The block header is constructed by six fields (**Table 2**), similar to the bitcoin block. The second field links the block to its parent block. All the transactions in the block are embedded into the header using a piece of data content, that is, the merkle tree root [28]. The merkle tree root assures the integrity of transactions as the alteration on transactions causes a totally different value of the merkle root value. Timestamp protects the block from time tampering. Without loss of generality, difficulty is a metric of how difficult it is to successfully find a hash. However, there are two distinct ways of describing the difficulty. The first describes it as the number of zeros at the start of the hash result of the block header, while the second one measures an estimated difficulty target. The target is the number of hash calculations to mine a block. An acceptable block must have a hash below this target level. We propose the same difficulty format as it in the bitcoin block, with the first two hexadecimal bits for the exponent and the remaining part is coefficient. Hence the target difficulty can be computed using (8) [27]:

$$target = coefficient \times 2^{8\times(exponent-3)}. \quad (8)$$

▼Table 2. The block format

Block Header	
Field	Description
Version	Block version number
Previous block hash	Hash of the previous block in the chain
Merkle tree root	Hash of the merkle tree root of transactions
Timestamp	Creation time of this block
Targeted difficulty	The proof-of-work difficulty target
Nonce	A counter for the proof-of-work

Block Payload (Transactions)	
Field	Description
Transaction 1	The first transaction in this block
…	…
Transaction *N*	The last transaction in this block

3.3.4 Mine Proof-Of-Work

In bitcoin, proof-of-work is a digital receipt which is hard to calculate but easy for others to verify [18]. A one-way cryptographic hash function, double SHA256, *dhash*(), is used to calculate the proof-of-work. This function is used in various fields of the bitcoin system [27], including the calculation of the merkle root. The result is calculated by hashing the candidate block header repeatedly, using different nonce value, until the resulting hash value matches the difficulty requirement. More specifically, the block is successfully mined if the hash result starts with the numbers of zeros. The number of zeros is equal to the difficulty.

To mine a block, each time a block candidate is released into SM network, and the hash of the block header is calculated by SMs. At the start of mining, a difficulty target is computed to get the maximum acceptable hash calculation times. An arbitrary number between 0 and the difficulty target is selected as the initial hash attempt number to start mining. As most of the proof-of-works does not appear within a small value of attempts. If it fails to find the proof-of-work within above the value range, the attempt value should starts from 0 to see if there is an answer among small numbers. When the total calculation times exceed the difficulty target, the SM fails to find a proof-of-work basing on this block. Therefore, the transactions must be rearranged and mined again. However, the mining work is aborted when the proof-of-work is found by someone else in SM network. **Algorithm 2** shows a summarised pseudocode of mining procedure.

Algorithm 2 Calculate Nonce (Proof-Of-Work)

Input: Candidate Block Header ***H***
Output: Nonce value nonce
1: Summarise the first five header fields in a basic string ***S***;
2: Calculate the difficult target *tar* using equation (**8**);
3: Initialise the tries number *nonce*, tried string *try*, output *result* from the double hash function *dhash()*;
4: Pick a random number **n** = *Random [0, tar)*;
5: *nonce* = **n**;
6: **while** (result is not found & *nonce* ⩽ *tar* & Not receive Proof-Of-Work from other SM) **do**
7: *result* = *dhash(try* + *nonce)*;
8: *nonce* ++ ;

9: **end while**
10: *nonce* = **0**;
11: **while** (result is not found & *nonce* ⩽ **n** & Not receive Proof-Of-Work from other SM) **do**
12: *result* = *dhash(try* + *nonce)*;
13: *nonce* ++ ;
14: **end while**
15: **if** (result is not found & Not receive Proof-Of-Work from other SM) **then**
16: **return** (*nonce* - 1);
17: **else**
18: Generate a new block header hash value by changing the sequence of transactions **then** Repeat the aforementioned steps;
19: **end if**
20: **End Algorithm**

4 Simulation and Evaluation

4.1 Assumptions

The assumed parameters are shown in **Table 3**. Our scenario is set to have each single RSU coverage range with 600 m and the maximum transmit power $P_{t-\max} = 20\,mW$ [29] in vehicles in the network simulation (Veins) [30]. VCS networks need decentralized management by RSU cells due to the fact that ITS application has to be employed in large scale geographical area. Therefore RSU in this scenario acts as the central key manager and a relay between vehicle nodes and the administrator agency. The 2^{10} vehicles pass an 8-row road area. The number of vehicles and rows are considered under a saturated traffic condition. The saturated traffic aims to exam our scheme under the worst case (as well as the heaviest burden of VCS). The vehicle speed follows normal distribution with μ = 46.56 and σ = 6.88 [25] while the departure time follows exponential distribution.

To improve rekeying efficiency, key tree structure of this scenario is based on LKH [8], [9] with binary tree degrees. The higher tree degrees result, the more node individual encryption upon rekeying. Batch rekeying is considered in the model with batch rekeying interval t_{BR} is set to 0.5 s. The benchmark BR scheme [10] is used. This scheme is the basic framework for all mobile networks. Even though there are some incremental schemes based on it, such as [7], but none of them are focus on VCS scenarios. Moreover, recent papers [11], [12] still use [10] as their basic idea.

We assumed that blocks are mined by Digilent Nexys-2 500 k that is considered as one of the lowest cost FPGA mining devices. This device can finish 5 million hash calculations per second. We take an average distance of 5000 m between SMs, while the distance between SM and RSU is set to 1000 m. The average transactions in a block is constrained by 2 and 128, which means the average vehicle departure requests a range by 2^{10} and 2^{7}. The range of the difficulty level is defined by 3 and 5.

▼Table 3. Assumption of scenario parameters

Parameter name	Parameter value
Length of RSU coverage area	600 m
BSM transmit power $P_{t-\max}$	20 mW
Overall vehicle number	2^{10} vehicles
Length of rekeying interval t_{BR}	0.5 s
Distance between SMs	5000 m
Distance between SM and RSUs	1000 m
Range of transaction numbers	2, 4, 8, 16, 32, 64, 128
Range of difficulty (the number of zeros)	3–5
Mining speed	5 million hashes per second

RSU: Road Side Unit SM: Security Manager

4.2 Key Initialisation

Table 4 presents the time cost for a vehicle to register to a RSU when it joins a new broadcast group. Results are generated in OMNeT++ 4.5 [30], [31]. The steps in the table follow the handshake routes in Fig. 3. Step 8 is a unique progress for batch rekeying, the central key manager collects all member list modification requests in this batch period and waits for the start of next batch interval. The rekeying message has complex format which contains information for all group members, therefore the processing time $t_{prepare}$ is much longer than other steps.

The vehicle sends IRM messages without any record about GK, therefore, it has to use its own public key to encrypt moving state information. ECIES with elliptic curve secp160r1 in Crypto++ [32] is selected for the cryptographic schemeECIES, and digital signature scheme ECDSA as well. The cipher block has a length of 75 bytes because ECIES provides much better

▼Table 4. Event timestamps

Step name	Timestamp
1. Vehicle joining	$t_0 = 0\ ms$
2. Registration Msg→RSU	$t_1 = 2.910098956\ ms$
3. RSU receives Msg	$t_2 = 3.040167479\ ms$
4. RSU checks Msg→PKI/CA	$t_3 = 4.350436255\ ms$
5. PKI/CA receives Msg (via router/switch)	$t_4 = 7.350735578\ ms$
6. PKI/CA checks Msg and sends toRSU	$t_5 = 7.351695577\ ms$
7. RSU receives Msg and prepares rekey	$t_6 = 7.372535577\ ms$
8. Send at next batch edge Wait time Rekey Msg preparation time	$t_{send} = t_{BR}$ $t_{wait} = t_{BR} - t_6$ $t_{prepare} = 4.289728099\ ms$
9. Send out rekey Msg	$t_7 = t_{BR} = t_{send}$
10. Vehicle receive rekey Msg	$t_8 = t_{BR} + 0.174698201\ ms$

CA: Certificate Authority PKI: Public Key Infrastructure RSU: Road Side Unit

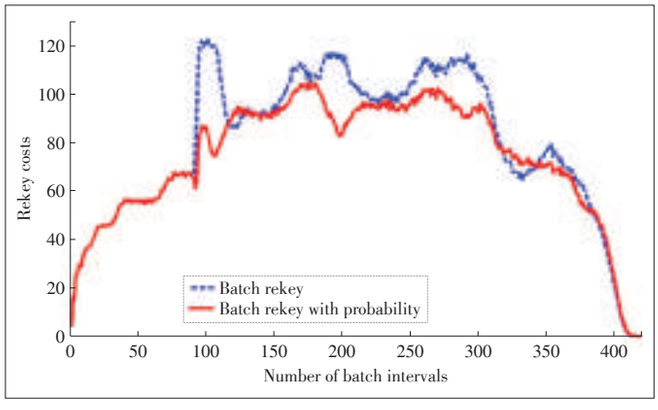
security level. The previous registered vehicle sends the normal BSM to inform RSU about region changing activity. The mobility state in BSM is encrypted by AES-CCM mode [33] by GK. The cipher text of AES has 32 bytes, which provides better efficiency. Digital signatures in both IRM and BSM are generated by ECDSA to demonstrate the authenticity of digital documents. In our scenario, the length of signature is 42 bytes, which provides authentication for messages.

4.3 Rekeying Costs

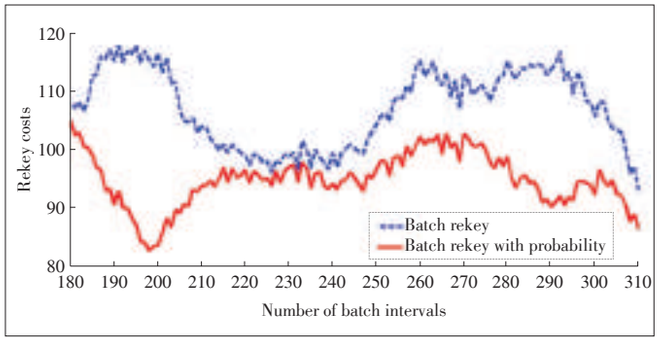
Our scheme is compared with the benchmark scheme from the aspect of rekeying costs, with reference to the batch interval number. To eliminate errors and generate a clear graph, 1000 times Monte Carlos simulations are used.

Fig. 7 demonstrates the rekeying costs of two schemes during a traffic flow of 2^{10} vehicles passing through. From the start to around the 80th batch interval, results are overlap to each other. The results are the same between the two schemes because the probability issue has not yet taken effect at the joining-only situation. Similar results are obtained after 350th interval.

The first node leaving activity happens in the 80th to 120th intervals. We can see that the probability based batch rekeying scheme has much better results when a node leaves suddenly, with approximately 33% less rekeying cost than the benchmark. More details about this region are shown in **Fig. 8**. The



▲Figure 7. Batch rekey costs for the complete simulation period.



▲Figure 8. Batch rekey costs for the stable phase.

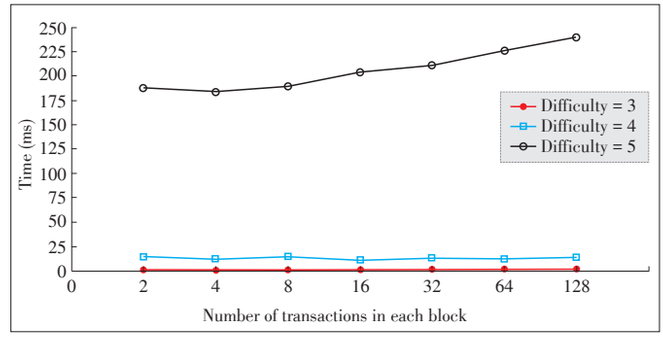
period from the 180th to 310th batch intervals is critical since both joining and leaving appear. Hence Most papers have focused on this period.

Fig. 8 presents more details about the results of the two schemes in the stable phase. The rekey cost for the benchmark algorithm has a sharp increase at about the 185th batch interval. A comparison of our proposed approach to the benchmark scheme shows that our scheme displays a more steady performance which means better robustness. The benchmark scheme shows a significant fluctuation which makes it difficult for the key manager to maintain the required Quality of Service (QoS) through the entire working period. In addition, the overall rekeying cost of our scheme is on average 18% less than that of the benchmark.

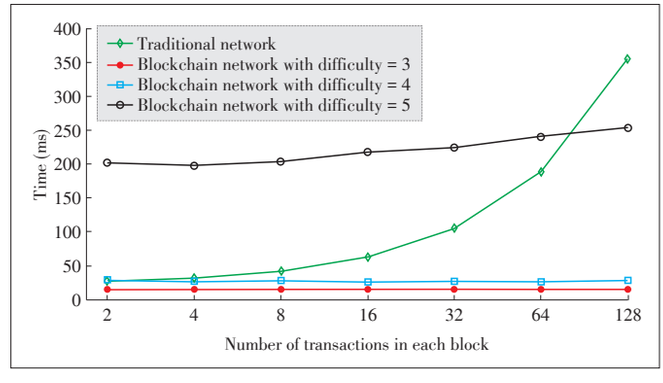
4.4 Key Transmission

Results of the mining time are compared in terms of mining difficulties. **Fig. 9** shows the mining time increases exponentially with the growth of difficulty. Mining runs in a short period of time when the level of difficulty equals to 3. The result of difficulty level 4 costs nearly double the time of difficulty 3 and their curves remain steady. However, difficulty level 5 costs nearly 8 times the time of difficulty 4 and the curve increases linearly.

Performance of key transmission is measured by the block propagation time from the current SM to destination SM. The overall handshake time cost in millisecond is shown in **Fig. 10**



▲Figure 9. Blockchain mining time.



▲Figure 10. Overall handshake time of two network structures.

in comparison with traditional VCS network structure. The handshake time of the standard network structure increases exponentially with an increasing number of transactions, which is due to the fact that CA must verify each transactions. The handshake time of the traditional network is much more than that of the blockchain network when difficulty is less than 5.

5 Conclusions

In this paper, we propose a key management scheme for group secure communication in heterogeneous VCS networks. Our scheme includes three components: group key management, key registration and key transportation. By simulating a vehicle group passing through different SM areas, our batch rekeying algorithm achieves more efficiency and robustness compared to the benchmark key management scheme. A faster key transmission time between the two security domains is presented with the help of blockchain.

For group key management, probabilities are introduced into the key manager so that the system can decide how to organise key tree properly. A model of vehicle registration is also discussed. The handshake presents the batch rekeying process. Our registration steps combine the registration messages with safety beacon messages that decrease overhead in the network. This procedure acts as foundation to implement further key management schemes. The blockchain concept is used to improve key transportation efficiency. Crossing border activities are formed into transactions and arranged into block. Third party central authorities are set aside since the verification job is delivered by SM network. The simulations show that the time cost for transporting keys is much less than that of standard network structure.

References

[1] P. Papadimitratos, L. Buttyan, T. Holczer, et al., "Secure vehicular communication systems: design and architecture," *IEEE Communications Magazine*, vol. 46, no. 11, pp. 100–109, Nov. 2008. doi: 10.1109/MCOM.2008.4689252.
[2] Y. Cao; N. Wang; G. Kamel; and Y. J. Kim, "An electric vehicle charging management scheme based on publish/subscribe communication framework," *IEEE Systems Journal*, vol. PP, no. 99, pp. 1– 14, 2015. doi: 10.1109/JSYST.2015.2449893.
[3] H. Hartenstein and L. P. Laberteaux, "A tutorial survey on vehicular ad hoc networks," *IEEE Communications Magazine*, vol. 46, no. 6, pp. 164– 171, Jun. 2008. doi: 10.1109/MCOM.2008.4539481.
[4] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, Jul. 2011. doi: 10.1109/JPROC.2011.2132790.
[5] *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Co-Operative Awareness Basic Service*, ETSI 102 637-2, 2010.
[6] *Dedicated Short Range Communications (DSRC) Message Set Dictionary*, SAE J2735, 2009.
[7] W. H. D. Ng, H. Cruickshank, and Z. Sun, "Scalable balanced batch rekeying for secure group communication," *Computers & Security*, vol. 25, no. 4, pp. 265–273, 2006. doi:10.1016/j.cose.2006.02.006.
[8] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 16–30, Feb. 2000. doi: 10.1109/90.836475.
[9] *Logical Key Hierarchy Protocol*, RFC2026, Mar. 1999.
[10] X. S. Li, Y. R. Yang, M. G. Gouda, and S. S. Lam, "Batch rekeying for secure group communications," in *Proc. ACM 10th International Conference on World Wide Web*, New York, USA, 2001, pp. 525–534. doi: 10.1145/371920.372153
[11] O. Zakaria, A. A. Hashim, and W. H. Hassan. "An efficient scalable batch-rekeying scheme for secure multicast communication using multiple logical key trees," *International Journal of Computer Science and Network Security (IJCSNS)*, vol. 15, no. 10, pp. 124–129, 2015.
[12] L. Veltri, S. Cirani, S. Busanelli, and G. Ferrari, "A novel batch-based group key management protocol applied to the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2724–2737, Nov. 2013. doi: 10.1016/j.adhoc.2013.05.009.
[13] K. Lu, Y. Qian, M. Guizani, and H. H. Chen, "A framework for a distributed key management scheme in heterogeneous wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 2, pp. 639–647, Feb. 2008. doi: 10.1109/TWC.2008.060603.
[14] Y. Sun, W. Trappe, and K. J. R. Liu, "A scalable multicast key management scheme for heterogeneous wireless networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 653– 666, Aug. 2004. doi: 10.1109/TNET.2004.833129.
[15] Y. Cao, Z. Sun, N. Wang, et al., "Geographic-based spray-and-relay (GSaR): an efficient routing scheme for DTNs," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1548–1564, Apr. 2015. doi: 10.1109/TVT.2014.2331395.
[16] X. B. Zhang, S. Lam, D. Y. Lee, and Y. R. Yang, "Protocol design for scalable and reliable group rekeying," *IEEE/ACM Transactions on Networking*, vol. 11, no. 6, pp. 908–922, Dec. 2003. doi: 10.1109/TNET.2003.820256.
[17] J. Daemen and V. Rijmen, *The Design of Rijndael: AES—the Advanced Encryption Standard*. Secaucus, USA: Springer-Verlag New York, 2002.
[18] S. Nakamoto. (2008, November). *Bitcoin: A peer-to-peer electronic cash system* [Online]. Available: https://bitcoin.org/bitcoin.pdf
[19] C. Decker and R. Wattenhofer, "Information propagation in the Bitcoin network," in *IEEE P2P 2013 Proceedings*, Trento, Italy, Sept. 2013, pp. 1–10. doi: 10.1109/P2P.2013.6688704.
[20] C. K. Wong and S. S. Lam, "Keystone: a group key management service," in *17th International Conference on Telecommunications*, Doha, Qatar, Apr. 2000.
[21] *IEEE Draft Standard for Wireless Access in Vehicular Environments (Wave)—Security Services for Applications and Management Messages*, IEEE P1609.2/D15, May 2012.
[22] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*. New York, USA: Springer-Verlag New York, 2004.
[23] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *International Journal of Information Security*, vol. 1, no. 1, pp. 36–63, Jan. 2001. doi: 10.1007/s102070100002.
[24] Y. H. Park, D. H. Je, M. H. Park, and S. W. Seo, "Efficient rekeying framework for secure multicast with diverse-subscription-period mobile users," *IEEE Transactions on Mobile Computing*, vol. 13, no. 4, pp. 783–796, Apr. 2014. doi: 10.1109/TMC.2013.40.
[25] M. R. Hustim and M. Isran. (2015, Dec 12). *The vehicle speed distribution on heterogeneous traffic: space mean speed analysis of light vehicles and motorcycles in makassar-indonesia* [Online]. Available: http://repository.unhas.ac.id/handle/123456789/16562
[26] W. D. Kelton and A. M. Law, *Simulation Modeling and Analysis*. Boston, USA: McGraw Hill Boston, 2000.
[27] A. M. Antonopoulos, *Mastering Bitcoin: Unlocking Digital Crypto-Currencies*. Sebastopol, USA: O'Reilly Media, Inc., 2014.
[28] R. C. Merkle, "A digital signature based on a conventional encryption function," *Advances in Cryptology—CRYPTO' 87*, vol. 293, no. 6, pp. 369–378, 1987. doi. 10.1007/3-540-48184-2_32.
[29] *DSRC Message Communication Minimum Performance Requirements: Basic Safety Message for Vehicle Safety Applications*, SAE Draft Std. J 2945, Jul. 2007.
[30] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved IVC analysis," in *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 3– 15, Jan. 2011. doi: 10.1109/TMC.2010.133.
[31] A. Varga, "The OMNeT++ discrete event simulation system," in *Proc. Europe-*

an Simulation Multiconference (ESM' 2001), Prague, Czech Republic, Jun. 2001, pp. 65.
[32] W. Dai. (2015, July 7). *Crypto++ library 5.6.0* [Online]. Abailable: http:/www.cryptopp.com
[33] *Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)*, RFC 4309, Dec. 2015.

Manuscript received: 2016-04-21

Biographies

LEI Ao (a.lei@surrey.ac.uk) received his BEng degree in communication engineering at Harbin Institute of Technology, China and University of Birmingham, UK, in 2013, and the MSc degree in communication engineering at the University of York, UK, in 2014. He is currently working toward the PhD degree in communication engineering in the Institute of Communication Systems at the University of Surrey, UK. His research interests include security and privacy for vehicular networks and privacy protection for location based services. He is currently involved with EU-funded PETRAS Project on security and privacy in smart vehicles and location based services.

Chibueze Ogah (c.anyigorogah@surrey.ac.uk) received the BSc (Hons) in computer science from the Ebonyi State University, Nigeria in 2005. He received the MSc degree (Distinction) in computer network technology from the University of Northumbria at Newcastle, UK in 2011. He is currently a PhD candidate at the Institute for Communication Systems, University of Surrey, UK. He has been a laboratory assistant and lecturer at the Computer Science Department of Ebonyi State University, Nigeria since September 2007 and February 2012. His research interests include security and privacy in vehicular networks, and Cisco routing protocols. He is currently involved with EU-funded PETRAS Project on privacy in smart vehicles.

Philip Asuquo (p.asuquo@surrey.ac.uk) received his Bachelor's degree in computer engineering from University of Uyo, Nigeria and MSc in computer network technology from Northumbria University, UK. He is currently working towards his PhD in electronic engineering at the University of Surrey, UK. His research interest includes cyber security of critical infrastructures, smart grid and smart homes, intelligent transport systems (ITS) and wireless sensor network security. He is currently involved with EU-funded PETRAS Project.

Haitham Cruickshank (h.Cruickshank@surrey.ac.uk) received a BSc degree in electrical engineering from the University of Baghdad, Iraq, in 1980, and MSc in telecommunications from the University of Surrey, UK and a PhD in control systems from Cranfield Institute of Technology, UK, in 1995. He is a senior lecturer at the Institute of Communication Systems (Formerly Centre for Communication Systems Research, CCSR), University of Surrey. He has worked there since January 1996 on several European research projects in the ACTS, ESPRIT, TEN-TELECOM, and IST programmes. His main research interests are network security, satellite network architectures, VoIP, and IP conferencing over satellites. He is a member of the Satellite and Space Communications Committee of the IEEE Communications Society, and is also a Chartered Electrical Engineer and IEE corporate member in the UK. He is active in the ETSI BSM and the IETF MSEC groups.

SUN Zhili (z.sun@surrey.ac.uk) received his BSc in mathematics from Nanjing University, China and PhD from the Department of Computing, Lancaster University, UK, in 1991. He is a professor at the Institute of Communication Systems (Formerly Centre for Communication Systems Research, CCSR), University of Surrey, UK. His research interests include wireless and sensor networks, satellite communications, mobile operating systems, traffic engineering, Internet protocols and architecture, quality of service, multicast, and security. He has been principal investigator and technical coordinator in a number of projects within the European Framework Program including the ESPRIT BISANTE project on evaluation of broadband traffic over satellite using simulation, the TEN-telecom VIPTEN project on QoS of IP telephony over satellite, the GEOCAST project on IP Multicast over satellites and ICEBERGS project on IP based Multimedia Conference over Satellite of IST, the SATELIFE project on Satellite Access Technologies on DVB-S and DVBRCS, and EuroNGI on next generation Internet.

DOI: 10.3969/j. issn. 1673-5188. 2016. S0. 006
http://www.cnki.net/kcms/detail/34.1294.TN.20160630.1830.002.html, published online June 30, 2016

Password Pattern and Vulnerability Analysis for Web and Mobile Applications

LI Shancang, Imed Romdhani, and William Buchanan
(School of Computing, Edinburgh Napier University, Edinburgh EH10 5DT, Scotland, UK)

Abstract

Text-based passwords are heavily used to defense for many web and mobile applications. In this paper, we investigated the patterns and vulnerabilities for both web and mobile applications based on conditions of the Shannon entropy, Guessing entropy and Minimum entropy. We show how to substantially improve upon the strength of passwords based on the analysis of text-password entropies. By analyzing the passwords datasets of Rockyou and 163.com, we believe strong password can be designed based on good usability, deployability, rememberbility, and security entropies.

Keywords

password strength; security entropies; password vulnerabilities

1 Introduction

Although receiving plenty of criticism, the text-based passwords are still heavily used for authenticating web and mobile application users [1], [2]. Many research efforts have been made to protect user's password against attacks [3]. In recent, many password managers have been developed to help people to create/manage secure passwords with enough strength and easy to remember (e.g. Dashlane, Keepass, Lastpass). However, when using a password manager, at least a master password needs creating and remembering [4], [5].

A number of websites have recently been hacked and millions of user credentials were leaked online [6], [7]. In 2012, six millions of LinkedIn users' credentials were leaked. Actually, it was reported that in this case over 117 million user credentials were leaked on the Dark Web [8]. In 2015, the Sony Pictures was hacked and many confidential data were leaked, including 173,000 emails and 30,000 separate documents [9]. It was reported that in China 130,000 users' data were leaked via China's train ticketing site 12306 [10] in Dec. 2014. In 2015, a large number of websites (including 163.com, CSDN, TianYa, Duduniu, 7k7k, 178.com, Rockyou, and Yahoo) were hacked and over 100 million of user credentials were leaked online. We believe that investigating the leaked passwords will be helpful in improving the strength of passwords in real data sources.

Most of users have various passwords for different web or mobile application accounts. However, it is difficult to remember so many passwords for a user. Although mobile devices have increasingly been used, it is still difficult to run a password manager over mobile devices. Besides, unfriendly on-screen keyboards make it more challenging or inconvenient to type passwords with special symbols or mixed-case characters [4]. Many websites and mobile applications (apps) require users to choose complicate passwords (e.g., mixed-case letters, digits, special characters) and the authentication of passwords becomes more complicated. In this case, the text-password input interfaces (e.g. touchscreen virtual keyboards) are applied to protect users' passwords from malwares [4].

In this paper, we will investigate these leaked passwords to comprehensively identify the strength of passwords. Basically, we will focus on four features of the passwords: 1) Length of passwords, 2) variety of character types in a password, 3) the randomness of passwords, and 4) uniqueness of passwords. Mathematically, we will analyze the password entropy, guessing entropy, and Minimum entropy for passwords in the leaked passwords lists. A number of password analyzing tools, including John the Ripper, Hashcat, and the password analysis and cracking toolkit (PACT) will be used to analyze the password lists for password length, password entropies, character types, pattern detection of masks, and other password features.

2 Background and Previous Work

There have been many research efforts made for helping users to choose passwords. Measuring the strength of passwords is an important topic. In [1], a method for calculating password

entropy was proposed, which is based on the summarization of the distributions of passwords length, placements of character, and number of each character types. Yan et al. [2] filtered weak passwords by improving dictionary-based checking with 7 character alphanumeric passwords. The password quality indicator (PQI) was proposed to measure or evaluate the quality of passwords [3], [4].

2.1 Typical Hash-Based Login Systems

A typical website or mobile application login system contains the following four basic steps: 1) A user registers and uses an assigned password; 2) the password is hashed and stored in the database (but the plain-text password should never be written to the database); 3) when logging the system, the hash of the entered password will be checked against the hash saved in the database; 4) if the hashes match, the user will be granted access. The weakness in this system is that the passwords can often be guessed, forgotten, or revealed. To overcome this weakness, a stronger password should be created or additional authentication factors be used, such as physical token, digital certificates, one-time access code, etc. In many mobile app authentication systems, the mobile devices are increasingly used to enhance login security [11].

A lot of concerns have been raised on the security of web and mobile application login systems, however, we should bring awareness to the inherent weakness of such systems that are vulnerable to passwords cracking by considering the following situations:

- Most authentication of websites and mobile applications have not moved to stronger hash type. Many weak hash types, such as Unix DES, NTLM and single-round salted SHA1 are still in use.
- Existing password policies have led to exploitable predictability.
- Authentication systems with design flaws are vulnerable to pass-the-hash attacks, for example the websites like 163com, CSDN, Tianya (before 2014).
- Power graphics processing unit (GPU) can significantly speed up brute force attacks to weak hashes, or even for long password length.
- In some applications, the username and password combination are inadequate for strong authentication.

2.2 Password Attacks

A hacker can break passwords with many ways, out of which the following attacks are widely used:

1) Dictionary Attack

A hacker may use a password cracking dictionary (such as wordlist, dictionary, and password database leak) to find a password. The password dictionary is a very large text files that includes millions of generic passwords. The hacker may use higher performance computer or game graphics cards to try each of these passwords until find the right one [12].

2) Brute-Force Attack

The brute-force attack, or exhaustive password attack, is still one of the most popular password cracking methods. It tries every possible combination until it gets the password. In practice, the password space for all possible combinations might be huge, which makes the brute force attack very difficult to carry out.

3) Man-in-the-Middle Attack

The man-in-the-middle (MITM) attacks may be used for regular web/mobile apps logins. It is possible for an attacker to find out authentication requests/responses from the recorded network traffic, and then capture candidates for password related contents. Passwords attack dictionary can be built by choosing login information over highly popular websites.

Traditionally, strong passwords are created by the following methods:

1) Proper length of passwords. The length of passwords should be properly selected by balancing the user convenience and security. An eight-character password with a mix of numbers, symbols, and uppercase and lowercase can take at most months or years to crack. There is no minimum password length everyone agrees on, but in generally a password is required to be a minimum of 8 to 14 characters in length. A longer password would be even better [13].
2) Mix of numbers, symbols, uppercase and lowercase. It is very difficult to crack such password; the only techniques are to try huge number of combinations until find the right one. For an eight-character password there are 83^8 possible combinations and need 10 days and 2 hours to crack [13], [14].
3) Salt password and avoiding passwords listed in password cracking dictionary. The dictionary, wordlist, and password database are widely used in password cracking. In [12], a password cracking dictionary with a size of 15 GB has been released, which was used to successfully crack 49.98% of a password list with 373,000 passwords. To create a safer password, a better salting scheme is needed.
4) One-size-fits-all password. Most websites or mobile apps apply the one-size-fits-all approach to ensure that users choose strong passwords.
5) Outsourcing the security. It is a trend for most websites or mobile apps to outsource their security systems. This trend will seemingly continue.

In summary, it is not very difficult to create a strong password with proper length and a mix of many different types of characters. It is hard to guess such passwords due to its randomness. However, memorizing such a strong password is a problem. It is very difficult for most users to memorize a strong password created with a random password generator of websites and mobile applications. In creating a strong and memorable password, we need to think about how to avoid using something obvious with dictionary characters. For example, we can create a strong password based on a simple sentence like "I

live in 20 Colinton Road at Edinburgh. The rent is $500 each month." We can easily turn this simple sentence into a strong password by using the first letter or digit of each word, as "Ili20CR.Edi $5em", which is a memorable and strong password with mix of numbers, characters, symbols, and uppercase letter and lowercase. It may be hacked in at least 420,805,123,888,006 years [14].

3 Password Strength Metrics and Evaluation

Password strength measurements can help to warn users away from highly vulnerable passwords [15]. Many authentication systems of websites and mobile applications require passwords must be able to resist eavesdroppers and off-line analysis of authentication protocols run. In general, the security of passwords can be measured with password strength. Password strength is defined in terms of probability of a determined attacker discovering a selected users' password by an inline attack. The password strength is also a function of both the entropy of the password and the way unsuccessful trials are limited. Entropy is believed as a standard measure of security [5].

3.1 Password Entropy

Shannon entropy is a popular method to evaluate the security strength of a password, which is also used as password entropy. Assuming a finite variable X corresponds to n passwords set (p_1, p_2 ,..., p_n), the password entropy can be modeled with Shannon entropy as $H(X)$

$$H(X)=-\sum_{i=1}^{n} p_i \cdot \log_2(p_i) \tag{1}$$

where p_i denotes the occurrence probability of ith possible outcome. A password using lowercase characters can be represented as $log_2(26) \approx 4.7$ bits of entropy per character. For a password "iliveinedinburgh" would have an entropy value of about $4.7 \times 16 \approx 75$ bits.

The Shannon entropy is commonly used to measure the passwords. Some variants of entropy have recently been proposed to measure other features of passwords such as guessing entropy, Minimum Entropy, and relative entropy.

3.2 Guessing Entropy

The ability of passwords that resists against complete off-line attacks can be measured with Guessing entropy. Guessing entropy is a measure of the difficulty to guess the passwords in a login system [6]. If the values of $Y=sort_d(X)$ are sorted with decreasing probability, the guessing entropy of Y can be defined as

$$G(Y)=\sum_{i=1}^{n} i \cdot p_i \tag{2}$$

The guessing entropy is closely related to the average size of passwords. If a password has n bits guessing entropy, an attacker has as much difficulty in guessing the average password as in guessing an n bits random quantity [16].

3.3 Minmum Entropy

Since in some cases, the password strength cannot warn users away from reusing the same password because they are usually based on heuristics (e.g., numbers, password length, upper/lowercase, symbols). Minimum entropy is a way to estimate the strength of a password, which is defined as

$$H_{\min}(X)=-\min \log_2(p_i) \tag{3}$$

For example, a low strength password p_b has low minimum entropy ($H_{min}(p_b) = 1$). High minimum entropy ($H_{min}(X) = \alpha$) guarantees that with high probability the adversary will always need to use around 2^{α} guesses to recover the users' passwords. The Minimum entropy shows the resistance of offline password cracking attacks with high probability.

3.4 Password over Mobile Applications

Many mobile applications require password input and the authentication task over mobile platforms is more complicated by using full-size key-board. In some mobile applications, the inconveniences caused by an unfriendly interface can affect users to create/use strong passwords. An example is that more than 80% of mobile device users are using digit-only passwords [6]. In recent, a number of password generation methods have been developed for mobile applications. For example, the object-based password (ObPwd) has been implemented over Android platform for generating password from a user-selected object (e.g., pictures) [7].

4 Analysis of the passwords

In this section, we investigated the way people create their passwords from five aspects: length, character types, randomness, complexity, and uniqueness. We analyzed over 100 million leaked and publicly available passwords from several popular websites (Rockyou, CSDN, TianYa, 163com).

4.1 Password Length

Most of the passwords have length of 6 to 10 characters as shown in **Tables 1** and **2** that specify the percentage of the total analyzed passwords.

▼**Table 1. Analysis of password length (Rockyou.txt)**

Length	Percentage (%)	Number of items
8	20	2,966,037
7	17	2,506,271
9	15	2,191,039
10	14	2,013,695
6	13	1,947,798

▼Table 2. Analysis of password length (163com.txt)

Length	Percentage (%)	Number of items
8	23	1,159,984
7	17	973,951
6	17	870,857
9	16	822,077
10	11	572,185
11	7	399,021
12	2	141,935
13	1	69,776
14	1	58,601

From both Tables 1 and 2, we can find that 85% of passwords are between 8 to 10 characters long, which is pretty predictable. Around 50% of the passwords both in Rockyou and 163.com lists are less than eight characters. Few passwords have a length greater than 13. The main reason is that most websites and mobile apps require a maximum length of 8 and long passwords are difficult to be remembered.

4.2 Character Types

The diversity of the character types in passwords can be categorized into the following sets: number, uppercase, lowercase, and special-case.

The character-sets in Rockyou.txt and 163com.txt are shown in **Tables 3** and **4**.

The character-set analysis helps us understand the usability and security of passwords. It is good to consider three or more character types when creating a password. More than 80% Rockyou passwords had only one-character type (lowercase). In 163com, more than 88% of the passwords had only numeric passwords.

4.3 Randomness

In our investigation, we found that many of the usual culprits are used such as "password", "123456", "abc123", and city names. We also found that many passwords were apparently related to a combination: part of user names, city names, country names, etc. A few of these are very specific but there may be context to this in the sign up process.

We analyzed the mask of passwords in both Rockyou.txt and 163com.txt (**Tables 5** and **6**). Table 6 shows only 1% of all the passwords have the patterns matching the advanced masks and the majority is "string-digit" passwords that consist of a string with two or four digits.

4.4 Uniqueness

This uniqueness is about password sharing for different accounts. According to the analysis of many Chinese websites (12306, 163.com, 126.com, Tianya, CSDN, etc.), we found that many users are sharing the same passwords among their accounts in these websites.

▼Table 3. Analysis of password character-sets (Rockyou.txt)

Character-set	Percentage (%)	Number of items
loweralphanum	88	4,720,183
upperalphanum	06	325,942
mixedalphanum	05	293,432

▼Table 4. Analysis of password character-sets (163com.txt)

Character-set	Percentage (%)	Number of items
Numeric	58	2,931,867
loweralphanum	30	1,527,719
loweralpha	08	450,746
loweralphaspecialnum	00	38,913
mixedalphanum	00	26,097
upperalphanum	00	23,905
specialnum	00	15,614
loweralphaspecial	00	4830
mixedalpha	00	4353
upperalpha	00	3142
All	00	2172
upperalphaspecialnum	00	1722
mixedalphaspecial	00	550
Special	00	164
upperalphaspecial	00	133

▼Table 5. Analysis of password advanced masks (Rockyou.txt)

Advanced Masks	Percentage (%)	Number of items
?l?l?l?l?l?l?d?d	07	420,318
?l?l?l?l?l?d?d	05	292,306
?l?l?l?l?l?l?l?d?d	05	273,624
?l?l?l?l?d?d?d?d	04	235,360
?l?l?l?l?d?d	04	215,074

It is believed that the leakage of such websites as 12306 and Tianya are caused by the hit-the-library attack, in which leakage of users' privacy data is more like that by a hacker hitting the library behavior. The hit-the-library attack is used by hackers to collect username, password, and other private information. After generating the corresponding dictionary with collected information, that attacker is able to attempt another batch landing sites. By this way, the hacker can deal with almost any website login systems. If a user uses the same username and password as the master key to log on different sites, he facilitates himself but also provides convenience for hackers.

In the Sony leakage case, 92% of passwords were reused across both in "Beauty" and "Delboca login systems. Only 8% of identical passwords are used. In internet web and mobile ap-

▼Table 6. Analysis of password advanced masks (163com.txt)

Advanced Masks	Percentage (%)	Number of items
?d?d?d?d?d?d	14	727,942
?d?d?d?d?d?d?d?d	13	701,557
?d?d?d?d?d?d?d	13	692,425
?d?d?d?d?d?d?d?d?d	06	348,786
?d?d?d?d?d?d?d?d?d?d	04	244,521
?d?d?d?d?d?d?d?d?d?d	03	162,921
?l?l?l?l?l?l?l?l	02	117,516
?l?l?l?d?d?d?d?d?d	01	90,281
?l?l?l?l?l?l?l?l?l	01	78,441
?l?l?l?l?l?l?l	01	74,678
?l?l?d?d?d?d?d?d	01	71,890
?l?l?l?l?l?l?l	01	67,221
?l?l?l?l?l?l?l?l?l?l	01	66,316
?l?l?d?d?d?d?d?d?d	01	65,743
?l?l?l?d?d?d?d?d?d?d	01	61,386
?l?d?d?d?d?d?d?d	01	53,065

plications, many users are using the same emails as their login usernames, which increases the risks of password sharing.

5 Conclusion

In this paper, we analyze the strength of passwords and investigate the password leakages cases from the viewpoints of length, character types, randomness, complexity and uniqueness, which is expected to warn users away from highly vulnerable passwords.

References

[1] W. E. Burr and D. F. Dodson. (2016, May 11). *Electronic Authentication Guideline: Recommendations of the National Institute of Standards and Technology, 1.0.2 ed.* [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-63/SP80063V1.pdf

[2] J. Campbell, D. Kleeman, and W. Ma, "Password composition policy: does enforcement lead to better password choices? " in *ACIS 2006*, Adelaide, Australia, 2006, Paper 60.

[3] J. Campbell, D. Kleeman, and W. Ma, "The good and not so good of enforcing password composition rules," *Information Systems Security*, vol. 16, no. 1, pp. 2–8, 2007. doi: 10.1080/10658980601051375.

[4] R. Cisneros, D. Bliss, M. Garcia, "Password auditing applications," *Journal of Computing in Colleges*, vol. 21, no. 4, pp. 196–202, 2006.

[5] M.-H. Lim and P. C. Yuen, "Entropy measurement for biometric verification systems," *IEEE Transactions on Cybernetics*, vol. 46, no. 5, pp. 1065–1077, May 2016. doi: 10.1109/TCYB.2015.2423271.

[6] M. Jakobsson, E. Shi, P. Golle, and R. Chow, "Implicit authentication for mobile devices," in *4th USENIX Conference on Hot Topics in Security*, Montreal, Canada, Aug. 2009, pp. 9–9.

[7] M. Mannan and P. C. van Oorschot. (2016, May 11). *Passwords for both mobile and desktop computers ObPwd for Firefox and Android* [Online]. Available: https://www.usenix.org/system/files/login/articles/mannan.pdf

[8] MakeUseOf. (2016, May 11). *What you need to know about the massive Linkedin accounts leak* [Online]. Available: http://www.makeuseof.com/tag/need-know-massive-linkedin-accounts-leak

[9] S. Fitz-Gerald. (2016, May 11). *Everything that happened in the Sony leak scandal* [Online]. Available: http://www.makeuseof.com/tag/need-knowmassive-linkedin-accounts-leak

[10] E. Yu. (2016, May 11). *130K users' data leaked via China's train ticketing site* [Online]. Available: http://www.zdnet.com/article/130k-users-data-leaked-via-chinas-train-ticketing-site

[11] M. Sarrel. (2016, May 11). *Authentication via Mobile Phone Enhances Login Security* [Online]. Available: http://www.informationweek.com/applications/authentication-via-mobile-phone-enhances-login-security/d/d-id/1103017?

[12] E. Escobar. (2016, May 11). *How long to hack my password* [Online]. Available: http://www.quickanddirtytips.com/tech/computers/how-to-crack-a-password-likea-hacker?page=1

[13] B. Buchanan. (2016, May 11). *Encryption* [Online]. Available: http://asecuritysite.com/encryption/passes

[14] Random-ize. (2016, May 11). *How long to hack my password* [Online]. Available: http://random-ize.com/how-long-to-hack-pass

[15] J. Blocki. (2016, May 11). *Password strength meters* [Online]. Available: http://www.cs.cmu.edu/ jblocki/entropyAndMinimumEntropy.htm

[16] NIST. (2016, May 11). *Electronic authentication guideline (NIST Special Publication 800-63)* [Online]. Available: http://itlaw.wikia.com/wiki/NIST-Special-Publication-800-63

Manuscript received: 2016-05-21

Biographies

LI Shancang (s.li@napier.ac.uk), PhD, is a lecturer in Network Forensics in School of Computing at Edinburgh Napier University, UK. Over the last few years, he has been working on a few research projects funded by EU, EPSRC, Academic Expertise for Business (A4B), Technology Strategy Board (TSB), and industry. Based on these research projects, dozens of papers have been published. His current research interests include network forensics, security, wireless sensor networks, the Internet of Things (IoT), and lightweight cryptography over IoT.

Imed Romdhani (i.romdhani@napier.ac.uk) is an associate professor in computer networking at Edinburgh Napier University, UK. He received his PhD from the University of Technology of Compiegne (UTC), France in May 2005, and an engineering and a master degree in networking obtained respectively in 1998 and 2001 from the National School of Computing (ENSI, Tunisia) and Louis Pasteur University of Strasbourg (ULP, France). He worked extensively with Motorola Research Labs in Paris and authored 4 patents in the field of IPv6, multicast mobility and IoT.

William Buchanan (w.buchanan@napier.ac.uk) is a professor in the School of Computing at Edinburgh Napier University, UK, and a fellow of the BCS and the IET. He currently leads the Centre for Distributed Computing, Networks, and Security and The Cyber Academy, and works in the areas of security, cloud security, web-based infrastructures, e-crime, cryptography, triage, intrusion detection systems, digital forensics, mobile computing, agent-based systems, and security risk.

DOI: 10.3969/j. issn. 1673-5188. 2016. S0. 003
http://www.cnki.net/kcms/detail/34.1294.TN.20160603.1100.002.html, published online June 3, 2016

Design and Implementation of Privacy Impact Assessment for Android Mobile Devices

CHEN Kuan-Lin and YANG Chung-Huang
(Dept. Software Engineering and Management, National Kaohsiung Normal University, Kaohsiung, Taiwan 802, China)

Abstract

There are a lot of personal information stored in our smartphones, for instance, contacts, messages, photos, banking credentials and social network access. Therefore, ensuring personal data safety is a critical research and practical issue. The objective of this paper is to evaluate the influence of personal data security and decrease the privacy risks in the Android system. We apply the concept of privacy impact assessment (PIA) to design a system, which identifies permission requirements of apps, detects the potential activities from the logger and analyses the configuration settings. The system provides a user-friendly interface for users to get in-depth knowledge of the impact of privacy risk, and it could run on Android devices without USB teleport and network connection to avoid other problems. Our research finds that many apps announce numerous unnecessary permissions, and the application installing confirmation dialog does not show all requirement permissions when apps are installed first time.

Keywords

privacy impact assessment; privacy risk; personal information; Android permission; configuration settings

1 Introduction

The sales of smartphones reached 1.2 billion units in 2014 [1]. According to the data from International Data Corporation (IDC), the worldwide smartphone market grew 13% year over year in 2015 Q2 [2]. Particularly, Android dominated the market with an 82.8% share in 2015 Q2 [2], leaving its competitors iOS, Windows mobile OS and Blackberry far behind.

Smartphones have become widespread because of a wide range of connectivity options such as Wi-Fi, GPS, Bluetooth and near field communication (NFC). However, ubiquitous internet connectivity and availability of personal information such as contacts, messages, photos, banking credentials and social network access has attracted the attention of malware developers towards the mobile devices and Android.

Internet security threat reports say that there are too many apps containing malware. Symantec has analysed about 6.3 million apps in 2014, and there are more than one million apps that are classified as malware which included 46 new families of Android malware [3]. In addition, there are approximately 2.3 million suspect apps. Technically, they are not malware, but they display undesirable behaviour, such as bombarding the user with advertising.

In order to avoid malicious apps from the official Google Play, Google introduced a security service named Bouncer [4], which can quietly and automatically scan apps. Any found malicious apps or malware that may be detrimental to users, damage the system or tries to steal privacy information, will be removed from Google Play.

Although Google had done a good job of keeping malware out of the store, the mobile threat report published by Lookout Mobile Security in 2014 showed that Android mobile devices encountered 75% more malware than that in 2013 [5]. Therefore, it is necessary to find more detail information about system and apps to avoid using malicious apps and protect personal or privacy information from being stolen.

In this paper, we propose a privacy impact assessment (PIA) system on Android mobile devices. The proposed framework evaluates the Android security risks based on permission request patterns of applications and configuration settings by users, which aims to minimise privacy risks. We also scan the log messages by a logcat command in Android shell, which helps us know what potential activities are running.

The rest of the paper is organized as follows. The second chapter introduces related work. The third chapter is the literature review about background information. The fourth chapter describes the system architecture and assessment rules. The fifth chapter demonstrates the design and implementation of

This work was supported in part by the Ministry of Science and Technology of Taiwan, China under Grant No. MOST 102-2221-E-017-003-MY3.

PIA. The sixth chapter is the practical test results including system performance. The seventh chapter is the conclusion and future work.

2 Related Works

2.1 Risk Assessment for Permissions

Yang Wang et al. did a quantitative security risk assessment for Android permissions and applications called DroidRisk [6]. Its objective is to improve the efficiency of Android permission system. They used two data sets with 27,274 benign apps from Google Play and 1260 Android malware samples, extracted the name, category, and requested permissions of each app by a crawler, and found the most significant differences between benign apps and malware.

The results demonstrate that malware are likely to request more permissions than benign apps. Malware also request more dangerous permissions that can change the settings or use money-related services than benign apps. Yang Wang et al. also computed the risk levels for all Android permissions. **Table 1** shows the top 20 permissions with highest risk levels [6].

2.2 Android Custom Permissions

Custom permissions are simply permissions declared by third-party applications. They are often used to protect different application components for services and content providers. For example, if Alice wants to share service between her own Android apps, the intent-filter in app A can be used for pending request, and then app B could use the intent to call the correspond service. However, in this case any apps can use app A's service if they know the service's action name in the intent-filter. Therefore, developers define their own custom permissions to protect their application components for data sharing. Any other apps cannot access a component unless the custom permission is requested and granted.

However, there are some security issues with custom permission. It might leak user data such as online browsing history, user's in-app purchases and fake messages inserted via its app [7]. The vulnerability is talking about the custom permission's registered strategy. Custom permissions are always defined as "signature" protection-level in order to check whether the apps is signed with the same key or not, but it may be damaged by a malicious app which defines the same permission name with "normal" protection-level during "Race" [8]. If the malicious app is installed on an Android device before the benign app, the same permission name will be registered using a "first one wins" strategy. This scenario allows all third-party apps to access the component and the sharing data [9].

▼**Table 1. Top 20 permissions with highest risk levels**

Ranking	Permission Name
1	WRITE_APN_SETTINGS
2	RECEIVE_WAP_PUSH
3	WRITE_SMS
4	INSTALL_PACKAGES
5	READ_SMS
6	RECEIVE_SMS
7	SEND_SMS
8	DELETE_PACKAGES
9	BROADCAST_PACKAGE_REMOVED
10	RECEIVE_MMS
11	CHANGE_WIFI_STATE
12	WRITE_CONTACTS
13	DISABLE_KEYGUARD
14	KILL_BACKGROUND_PROCESS
15	READ_LOGS
16	CALL_PHONE
17	MOUNT_UNMOUNT_FILESYSTEMS
18	PROCESS_OUTGOING_CALLS
19	SET_WALLPAPER_HINTS
20	EXPAND_STATUS_BAR

3 Literature Review

3.1 Privacy Impact Assessment

A PIA is a process for evaluating a proposal in terms of its impact upon privacy, which helps an agency identify the potential effects [10]. PIA enables an organisation to systematically and thoroughly analyse how a particular project or system will affect the privacy of the individuals involved [11]. PIA aims to minimise privacy risks. With it, we can identify and record risks at an early stage via analysing how the purposed uses of personal information and technology will work in practice.

3.2 Android Permission Framework

Android apps can only access their own files by default. In order to interact with the system and other applications, such apps request additional permissions that are granted at the installed time and cannot be changed [8].

Android provides a permission-based security model in the application framework. Developers must declare the permissions required using the <uses-permissions> tag in AndroidManifest.xml [12]. Android permissions are divided into four protection-levels, with different potential risks as discussed [13]:

1) Normal: A lower-risk permission that gives requesting applications access to isolated application-level features, with minimal risk to other applications, the system, or the user. The system automatically grants this type of permission to a requesting application at installation, without asking for the

user's explicit approval.

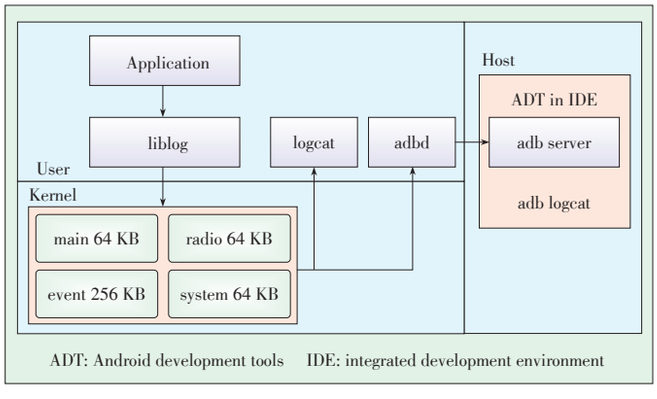
2) Dangerous: A higher-risk permission that gives a requesting application access to private user data or control over the device, which may cause negative impact on the user. Because this type of permission introduces potential risks, the system may not automatically grant it to the requesting application. A user must accept the installation of dangerous permissions at the install time.
3) Signature: A permission that the system grants only if the requesting application is signed with the same certificate as the application that declared the permission. If the certificates match, the system automatically grants the permission without notifying the user or asking for the user's explicit approval.
4) SignatureOrSystem: A permission that the system grants only to applications that are in the Android system image or that are signed with the same certificate as the application that declared the permission.

3.3 Android Logger

Logging is an essential component of any Linux operating systems, including embedded ones. Either post-mortem or real-time analysis of a system's logs for errors or warnings is vital to isolate fatal errors [14]. Though Android's kernel still maintains its own Linux-based kernel-logging mechanism, it also uses another logging subsystem, colloquially referred to as the logger. This driver acts as the support for the logcat, dmesg, dumpsys, dumpstate and burgeport command. One program logcat displays a continuously updated list of system and application debug messages [15]. It provides four separate log buffers, depending on the type of information: main, radio, event and system [16]. **Fig. 1** shows the flow of log event and components that assist the logger.

3.4 Android Intent

The primary method for inter-component communication, both within and between applications, is via intents [17]. It can be used with startActivity to launch an Activity, broadcastIntent to send it to any interested BroadcastReceiver components, and startService or bindService to communicate with a background Service [18].



▲Figure 1. Android log system.

3.5 Static and Dynamic Analysis

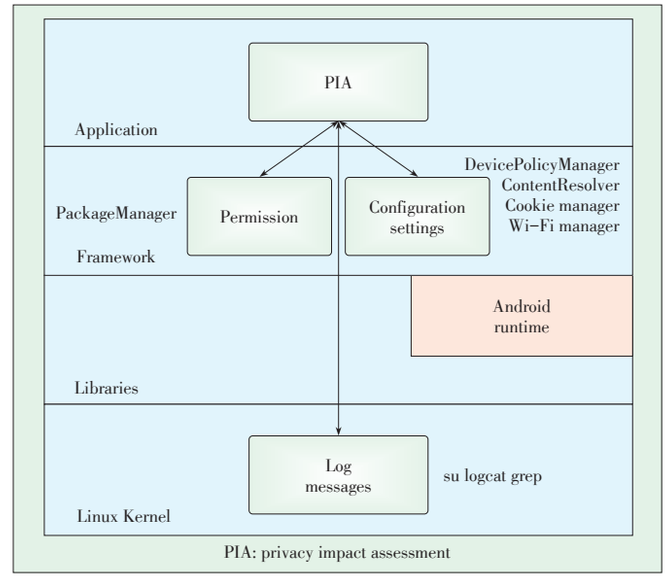
Static analysis works by disassembling and decompiling without actually running application. This does not affect the device. The methods for static analysis are quick, but they fail against the encrypted, polymorphic and code transformed malware [12]. Dynamic analysis methods execute an app in a protected environment, emulating all the resources and features. Therefore, they inspect its interaction for identifying malicious activities.

3.6 CIS Security Benchmark

The Security Benchmark [19] was proposed by the Center for Internet Security (CIS) and is a standard for security configuration of operating systems, including Linux, Windows, iOS and Android. This document defines the situation where operating system configuration settings should be in order for the system to be more secure. System administrators or users can set an Android operating system configuration based on this document in order to heighten the security of Android mobile devices.

4 System Architecture

A PIA system is designed for Android mobile devices, which supports the versions of Android above five. The PIA system can be installed on Android platforms because it is implemented into an Android application package (APK), and it does not need any additional condition such as network connection or USB teleport, which avoids other privacy risks. The system is composes of three parts as shown in **Fig. 2**. It per-



▲Figure 2. PIA system architecture.

forms with static method and dynamic methods at the same time.

4.1 Identifying Permission Requirements

The PIA system invokes the package manager to parse each app's permissions which should be declared in the AndroidManifest.xml file by developers (**Fig. 3**), and then calculates the privacy impact assessment score automatically referring to the protection-level [13] defined by Google and the report of the top 20 permissions with highest risk levels [6]. This main purpose of this method is to detect whether the app is using excessive permissions for dangerous requirements.

In general, the privacy impact assessment score of permissions for different categories are listed in **Table 2**. But, if the permission is in the top 20 with highest risk levels, the privacy impact score must be raised up to 10, because it can cause more huge damage to the system or the user and the app may have horrible potential motivation that declares the permission with highest risk levels. A custom permission defined by developers aims to share data or components with the developers' applications, so its privacy impact score is 8. The custom permission still has some potential privacy risks discussed in section 2 even if it can generally protect other apps to access data

4.2 Analysing Configuration Settings

According to the configuration of the CIS Security Benchmark document [19] (**Table 3**), the PIA system verifies the items and configuration type by ContentResolver, device policy, cookie and Wi-Fi manager, in order to improve and repair configuration settings of the Android mobile device. If the system configuration does not pass the test, its privacy impact score is eight point, in contrast, its privacy impact score is zero point. The main purpose of this method is to suggest users what system configurations they should adjust and then actually enhance the security of the device with international safety standards.

```
//getPermission
try {
    PackageInfo pinfo =
      getPackageManager().
        getPackageInfo
          (pkg, PackageManager.GET_PERMISSIONS);
    group = pinfo.requestedPermissions;
}
```

▲**Figure 3. Getting requested permissions by package manager.**

▼**Table 2. Privacy impact score**

Permission type	Privacy impact score	Top 20 with highest risk level
Normal	2	10
Dangerous	8	10
Signature	5	–
SignatureOrSystem	5	–
Custom	8	–

4.3 Detecting Potential Activities

In order to mine characters related to the Android intent, the PIA system uses su and logcat command with grep command together in the Android shell, which aims to track the messages stored in the log buffer. This method can scan potential events or harmful activities when apps installed in the device are running. **Fig. 4** shows the source code of the system that filters a specific keyword in the Android shell.

4.4 PIA Score Formula

The system finally computes the total privacy impact scores referring to Tables 2 and 3, according to the following formulae:

1) The App's PIA score: The sum of all permission scores is divided by the quantity of permission.

▼**Table 3. Verifing items**

Verifing item	Type	Pass/fail
Android version	System	0/8
Set auto-lock time	System	0/8
Third-party apps	System	0/8
Set screen lock	System	0/8
Encrypt phone	System	0/8
Disable Wi-Fi	Device	0/8
Disable camera	Device	0/8
Browser cookie settings	Browser	0/8

```
//su
Process p = Runtime.getRuntime().exec("su");
DataOutputStream pp =
  new DataOutputStream(p.getOutputStream());

//logcat
pp.writeBytes("logcat -v time -d |grep ''
  + package_name
    + "' |grep 'android.intent.action.'\n");

BufferedReader bufferedReader =
  new BufferedReader
    (new InputStreamReader(p.getInputStream()));
pp.writeBytes("exit\n");
pp.flush();

StringBuilder log = new StringBuilder();
String line;
while ((line=bufferedReader.readLine())!=null)
{
    log.append(line);
    log.append("\n");
}
```

▲**Figure 4. The source code of the system that filters a specific keyword in the Android shell.**

2) The Android system's final PIA score: The sum of all App scores plus the sum of the configuration scores is divided by the quantity of App plus the quantity of the configuration item.

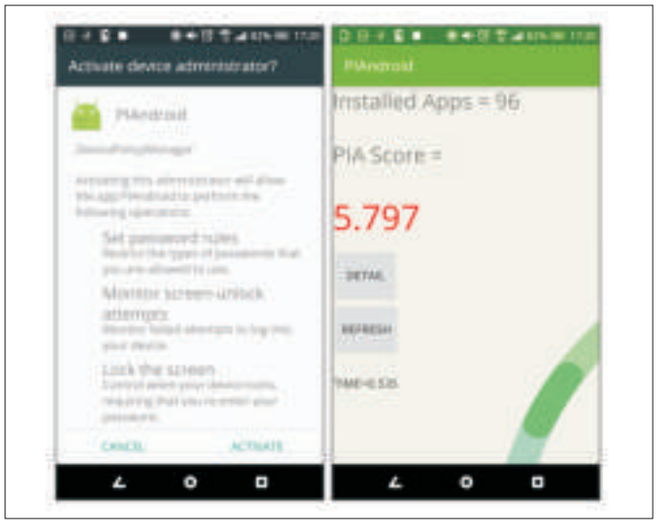
As a result, the user is able to get in-depth knowledge of the impact of privacy risks on each Android mobile device.

5 System Implementation

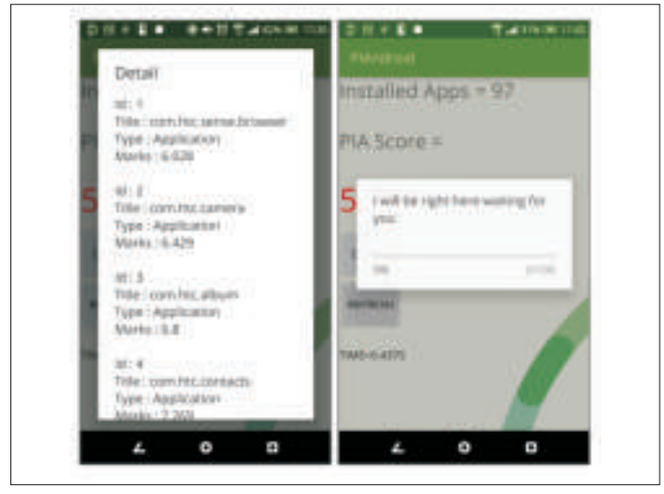
After installing an app on an Android mobile device, the user can begin to use it. When the app is executed, the screen in **Fig. 5** will appear. At this time, the user must activate the device administrator for the system because the device policy manager will be used in the setting page in order to repair system configurations. Then the user can observe the Android system's privacy impact assessment result in the home page. When the user clicks the "detail" button, the page will show the detail information that including each app's score and system configuration score. If the user click "refresh" button, the system will calculator the PIA score again in the background thread by AsyncTask (**Fig. 6**).

The "permission" page displays all apps that have been installed in the mobile device shown (**Fig. 7**). If the user selects any apps in the list, he can enter the interface shown in **Fig. 8** and read all permissions that the apps request, including normal, dangerous, signature, system and custom permissions. The user can also check the introduction about permission's protection-level information by click "introduce" button.

In the "intent" page, the user can click the "monitor" button to start scanning log messages from the log buffer by su, logcat and grep commands. The screen displays the record related to the Android intent or Apps' activities. For example, when the Google drive uploads a photo from the user's device, the "android.intent.action.SEND" log message will appear in the log buffer, and then the system shows "sending data" on the screen. However, if the user does not have super user privilege, which means he has not rooted his device, the screen will display "sorry" message as shown in **Fig. 9**.



▲Figure 5. Activating the device administrator and the home page.



▲Figure 6. Detail information and refreshing.

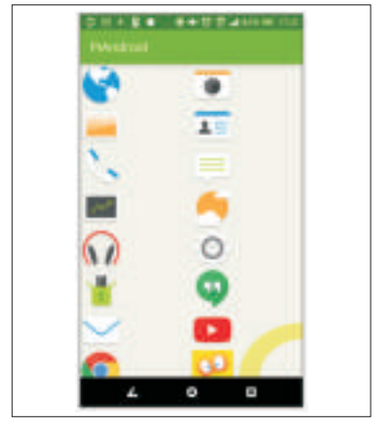
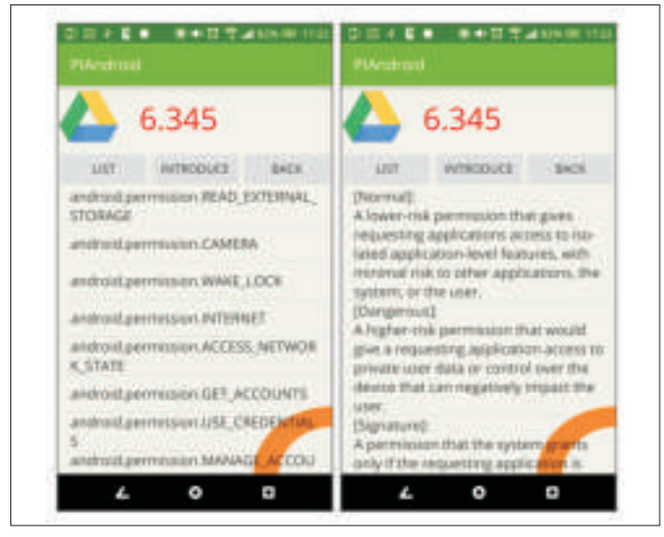
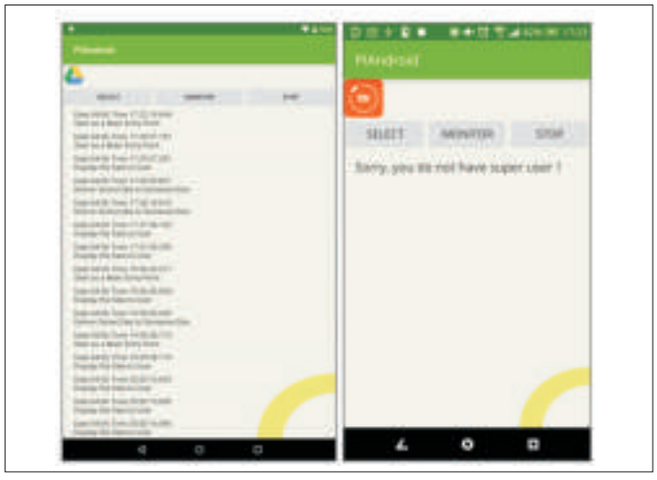


Figure 7. ▶ List of all applications.



▲Figure 8. List of all permissions and the protection-level introduction.



▲Figure 9. Display of log messages.

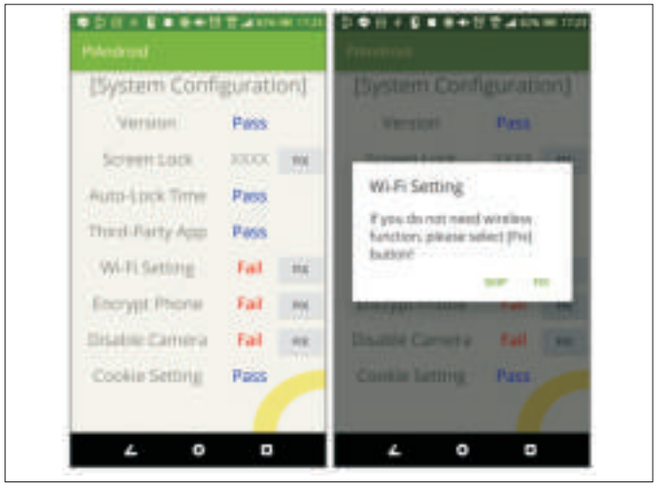
In the "setting" page, the user sees the interface shown in **Fig. 10**. The items shown on the left column are the test items, while the middle column shows the test results. The "FIX" button shown on the right column appears when the device does not pass the test.

After clicking the "FIX" button, the user chooses to fix the setting or skip this item, if the user chooses fixing, repair will be done automatically or the interface jump to the settings page.

6 Practical Tests

In our PIA system, the application installing confirmation dialog did not show all requirement permissions such as "Signature", "SignatureOrSystem" and custom permissions when apps were installed first time,. Many apps announce numerous unnecessary permissions, for example, Mi Fit is used to connect with Mi band to set, track and follow the user's health and fitness data, but this app announces the "camera" permission and "read external storage" permission.

The system performance was assessed, including the execution time, CPU and memory used. We marked the passing time of the system that calculated the PIA score when it ran on the device first time, and used the CPU monitor to record CPU and memory used. The test environment and results are shown in **Table 4** and **Fig. 11**.



▲Figure 10. Detecting and repairing system configurations.

7 Conclusions and Future Work

The privacy risk of personal information is a serious issue to mobile device users when they use various apps for work or leisure in their daily life. In this paper, a privacy impact assessment framework for Android mobile device offers the management of privacy risks, including apps' permission, app execution information and unsuitable system configurations. After scanning is completed and results are obtained, the final report can be used for information security verification, when an app's score on the report is too high, the user can choose not to install the app to make his device more reliable. While a failed item on the setting page needs to be corrected, the user can click "FIX" button to make corrections. The purpose of this system is to reduce, eliminate and minimize privacy risks to the user or vulnerabilities. It improves the security of mobile devices.

The research contribution of this study lies in developing a

▼Table 4. System performance

Device	Nexus 7	HTC E8
Version	6.0.1	6.0.1
Quantity of apps	30	97
Average time (s)	0.755	0.552
Max CPU used	9%	27%
Memory used (MB)	162	171

▲Figure 11. CPU and memory used.

remediation system for Android platforms. Users can run the system on their devices by installing this app, and the system does not need any additional conditions such as internet connection or standard USB teleport, which especially avoids other risks. However, this study plans to break through the restriction of the shell command, because it is too dangerous to use su command for Android system. This may cause more disadvantage; after gaining root privilege, an app has access to the entire system and to low-level hardware. Malicious apps can abuse su to allow themselves irremovable, bypass Android's security measures, and infect smartphones system [20].

References

[1] Gartner. (2015, March 3). *Gartner Says Smartphone Sales Surpassed One Billion Units in 2014* [Online]. Available: http://www.gartner.com/newsroom/id/2996817
[2] International Data Corporation. (2015, August). *Smartphone OS Market Share, 2015 Q2* [Online]. Available: http://www.idc.com/prodserv/smartphone-os-market-share.jsp
[3] P. Wood, B. Nahorney, K. Chandrasekar, et al., "Internet security threat report," Symantec, Mountain View, USA, Tech. Rep. vol. 20, Apr. 2015.
[4] N. J. Percoco and S. Schulte, "Adventures in bouncerland: failures of automated malware detection within mobile application markets," in *Black Hat*, Las Vegas, USA, Jul. 2012.
[5] Lookout Mobile Security, "2014 mobile threat report," Lookout, San Francisco, USA, Tech. Rep, 2014.
[6] Y. Wang, J. Zheng, C. Sun, and S. Mukkamala, "Quantitative security risk assessment of android permissions and applications," in *27th International Conference on Data and Applications Security and Privacy XXVII*, Newark, USA, Jul. 2013, pp. 226–241. doi: 10.1007/978-3-642-39256-6_15.
[7] TrendLabs Security Intelligence Blog. (2014, March 20). *Android Custom Permissions Leak User Data* [Online]. Available: http://blog.trendmicro.com/trendlabs-security-intelligence/android-custom-permissions-leak-user-data/
[8] P. Walvekar. (2014, April 16). *Race Conditions on Android Custom Permissions* [Online]. Available: https://datatheorem.github.io/2014/04/16/custom-permissions/
[9] N. Elenkov, *Android Security Internals: An In-Depth Guide to Android's Security Architecture*. San Francisco, USA: O'Reilly Media, 2014.
[10] *Privacy Impact Assessment Handbook*, Office of the Privacy Commissioner, Wellington, New Zealand, 2007, pp. 5, 21–27.
[11] Information Commissioner's Office, "Conducting privacy impact assessments code of practice," ICO, Scotland, UK, 2014.
[12] P. Faruki, A. Bharmal, V. Laxmi, et al., "Android security: a survey of issues, malware penetration and defenses," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 998–1022, Dec. 2015. doi: 10.1109/COMST.2014.2386139.
[13] Developer Console. (2016). <*permission*> [Online]. Available: http://developer.android.com/intl/zh-tw/guide/topics/manifest/permission-element.html
[14] K. Yaghmour, *Embedded Android*. San Francisco, USA: O'Reilly Media, 2013.
[15] A. Hoog, *Android Forensics: Investigation, Analysis, and Mobile Security for Google Android*. Waltham, USA: Elsevier, 2011.
[16] J. J. Drake, P. O. Fora, Z. Lanier, et al., *Android Hacker's Handbook*. Indianapolis, USA: John Wiley & Sons, 2014.
[17] W. Klieber, L. Flynn, A. Bhosale, L. Jia, and L. Bauer, "Android taint flow analysis for app sets," in *3rd ACM SIGPLAN International Workshop on the State of the Art in Java Program Analysis*, New York, USA, Jun. 2014, pp. 1–6. doi: 10.1145/2614628.2614633.
[18] Developer Console. (2016). *Intent* [Online]. Available: http://developer.android.com/intl/zh-tw/reference/android/content/Intent.html
[19] Center for Internet Security. (2012, October 1). *CIS Google Android 4 Benchmark v1.0.0*. [Online]. Available: https://benchmarks.cisecurity.org/downloads/show-single/index.cfm?file=android4.100
[20] Y. Shao, X. Luo, and C. Qian, "RootGuard: protecting rooted android phones," *IEEE Computer Society*, vol. 47, no. 6, pp. 32–40, Jun. 2014. doi: 10.1109/MC.2014.163.

Manuscript received: 2016-04-21

Biographies

CHEN Kuan-Lin (kuanlin81625@outlook.com) received the BS degree from National Pingtung University of Education in 2014. He is currently a master student at Department of Software Engineering and Management, National Kaohsiung Normal University. He is actively involved in mobile platform security.

YANG Chung-Huang (chyang@nknu.edu.tw) has a PhD degree in computer engineering from the University of Louisiana at Lafayette in 1990. He is currently professor at the National Kaohsiung Normal University, distinguished professor at the Xi'an University of Posts and Telecommunications, and guest professor at the Xidian University. Previously, he was a software engineer at the RSA Data Security, Inc. (Redwood City, USA) in 1991, a postdoctoral fellow at the NTT Network Information Systems Laboratories (Yokosuka, Japan) in 1991-1993, and a project manager of the Information Security and Cryptology Project at the Telecommunication Laboratories, Chunghwa Telecom (Taiwan) in 1995-1997. For more details, please refer to http://security.nknu.edu.tw/.

DOI: 10.3969/j. issn. 1673-5188. 2016. S0. 005
http://www.cnki.net/kcms/detail/34.1294.TN.20160622.1830.002.html, published online June 22, 2016

SeSoa: Security Enhancement System with Online Authentication for Android APK

DONG Zhenjiang[1], WANG Wei[1], LI Hui[2], ZHANG Yateng[2], ZHANG Hongrui[2], and ZHAO Hanyu[2]
(1.Cloud Computing and IT Research Institute, ZTE Corporation, Nanjing 210012, China;
2.Beijing University of Posts and Telecommunications, Beijing 100876, China)

Abstract

Android OS provides such security mechanisms as application signature, privilege limit and sandbox to protect the security of operational system. However, these methods are unable to protect the applications of Android against anti-reverse engineering and the codes of such applications face the risk of being obtained or modified, which are always the first step for further attacks. In this paper, a security enhancement system with online authentication (SeSoa) for Android APK is proposed, in which the code of Android application package (APK) can be automatically encrypted. The encrypted code is loaded and run in the Android system after being successfully decrypted. Compared with the exiting software protecting systems, SeSoa uses online authentication mechanism to ensure the improvementof the APK security and good balance between security and usability.

Keywords

software protection; anti-reverse; Android; authentication

1 Introduction

Because of its true openness, Android [1] has become one of the most popular mobile platforms in the world since its appearance in the market a few years ago. Naturally, the number of available applications is increasing rapidly and now over 1,800,000 Android applications can be chosen by the Android users according to the statistical result of Google Play [2]. However, the open source code of Android and a huge number of applications pose a vital security challenge for the Android system. More and more attackers aim to its flaws, not only the flaws of the OS itself, but also the flaws of the applications, which are hard for the developers of Android system to fix.

Among many security threats of Android system, the Android applications' source codes face the risk of being decompiled and may be acquired by illegal organizations through reverse engineering techniques. This threat may result in that the program is slightly modified, re-packaged and released in the form of new applications; even worse, attackers may tamper with the code, mix malware to obtain the user's sensitive data, steal his bank account numbers and passwords stored on the phones, or to increase the charges by triggering some pay-needed operations. It is difficult for the Android existing security mechanisms to detect and prevent this kind of threats, which may make users suffer from certain direct or indirect economic losses.

To avoid the above mentioned threats, researches have proposed some mechanisms to protect the software against anti-reverse. One of the solutions is code obfuscation [3], which can be used to transform the original code into a form that makes reverse engineering harder and more time consuming and at the same time still possess the same function as the original software has. In general, the code obfuscation techniques include renaming of the identifiers of the variables, constants, classes, methods, etc. in the program. Such an obfuscation tool is ProGuard [4], which is integrated into the Android software development kit. In [5], a modification of Tiny C Compiler (TCC) , a simple compiler, is proposed to modify certain unconditional branching instructions to conditional branching and to make confusing conversion on the critical data of the software. This modification aims at misleading automated reverse engineering tools [6] to detect the original code. However, code obfuscation just makes a simple change of the names of classes or variables. If only this method is used to protect Android apps, as long as the .dex files are found, they can be decompiled into .smali or .java, and then be reversed.

To protect the software more effectively, new tools as DexGuard [7] and Allatori [8] appear, which also support a number

This work was supported by National Natural Science Foundation of China (61370195) and ZTE Industry-Academia-Research Cooperation Funds.

of other features such as control flow randomization, string and class encryption, and temper detection. In 2014, Piao [9] reveals that DexGuard has some weaknesses and an attacker is able to analyze the hex code of a Dalvix executable file. Piao suggests to store the core execute class file through obfuscation on the server; in this way, when an application needs to execute core routines, it must request these routines from the server, download it and maybe decrypt it. Our analysis shows that Piao's solution requires the availability of server all the time while the protected Android application is running. To overcome this shortage, a security enhancement system with online authentication for Android application package (APK) called SeSoa is proposed in this paper, in which Android applications are only authenticated by the authentication server before being run for the first time after they are downloaded and installed, which reduces the burden on the server.

The rest of this paper is organized as follows. In Section 2, related work is discussed and the existing security problems related with application protection in the Android security model and the security goals in SeSoa are analyzed. In Section 3, a novel solution of protecting software in Android platform is provided. In Section 4, the security of the proposed solution is analyzed. And finally, the conclusions are presented in Section 5.

2 Related Work

2.1 Problems in Android Security Model

Because of the open-source feature, Android requires strict security specification and robust security architecture. The security feature is reflected in the system security design structure, including all aspects in the application framework layer and the core layer. At the application layer, there are signature mechanism and application access control mechanism to protect the application security. At the kernel level, Android obtains its security goals based on the security features of Linux kernel systems. Therefore, the resources of different processes are well isolated by the sandbox mechanism, the unique memory management mechanism and the efficient and safe inter-process communication mechanism. Such security mechanisms provided by the Android system achieve the security goals to some extents, but they, especially Android signature and sandbox mechanisms, fail to protect the software of application from anti-reverse.

Firstly, Android system tries to protect the security of the application using the signature mechanism, which means that the applications installed in Android system are required to be signed by some institutes. In fact, the signature mechanism is not just used to identify the application's developer, it is also used to detect whether the application has been illegally tampered with. If the answer is yes, the program's signature cannot pass the verification of Android system, so the result is that the tempered application does not run correctly. Android system allows application with self-signed, which means that the applications can be signed by the developers themselves. If the signed APK package is decompressed, a folder named META_INF will be found, in it is the signature information of the applications. The folder contains MANIFEST.MF, CERT.SF and CERT.RSA files, multiple RSA file appears if multiple certificate signing is used. MANIFEST.MF is the main APK summary information, such as the APK information, application properties and the hash values of all files. CERT.SF is the signature file obtained using SHA1withRSA signature algorithm, which contains a summary of the application signature value. Attackers often get the correct signature information for the software by analyzing the file and tamper the software illegally, regenerate the signature and rewrite these files to pass the verification of the Android system.

Secondly, since Android is a multi-process system, isolating the resources for each application is a basic requirement for security. The sandbox mechanism is adopted in Android system to make sure that every application's process is a secure sandbox running in its own instances with a unique ID (uID) assigned to it. With such a mechanism, each application is running in a separate Dalvik virtual machine (DVM), with a separate address space and resources. As DVM is running on the Linux process and dependent on the Linux kernel , Android uses DVM and Linux file access control to achieve the sandbox mechanism. Any application that wants to access for system resources or other application permission must be declared in its manifest files or shared uID. But this sandbox isolation technology on Android also makes the code of its application face the threat of being decompiled. The reason is that the Java language application needs to be compiled into a binary byte code, which is the intermediate code running on DVM and can more easily be decompiled based on the Java decompile technology, be reverted from the original code to the logic results and get the identifiers names or other information.

In addition, due to the open source feature of the Android platform, the decompile technology for the applications on Android platform has been fully studied, thus through a number of mature disassembly tools, it is not difficult to get the Smali code of the software and then the source code through the reverse analysis. This makes the study on how to protect the code effectively in Android platform very important.

2.2 Goals of Application Protection

Generally, a software protection system should meet the following secure requirements.

1) Anti-tamper, preventing the application from being modified by some attackers
2) Preventing dynamic debugging, i.e., preventing attackers from getting the source code of application by using dynamic debugging tools;
3) Preventing decompilation, i.e., preventing attackers from getting the source code of application by using decompile

tools.

3 Proposed Solution

3.1 SeSoa Overview

To meet the above-mentioned requirements, a new solution called security enhancement system with online authention (SeSoa) is proposed. The main idea is using an authentication server to verify whether the APK is enhanced by the authorized party and integrity of the APK. The SeSoa consists of the operator, authentication server (AuS), security enhancement server (SeS) and Android client equipment. The main security reinforcement software is running on the SeS to generate templates packers, encrypt the APK's core files, regenerate signature of APK through implement the script file, and repackage the new APK. The authentication server is used to verify the integrity of the downloaded APK at the Android client side and source of APK. The Android client is mainly responsible for running the environmental monitoring, integrity verification and decryption of encrypted part of code and loading the original APK code. **Fig. 1** shows the SeSoa system and its operational flows.

1) Input APK

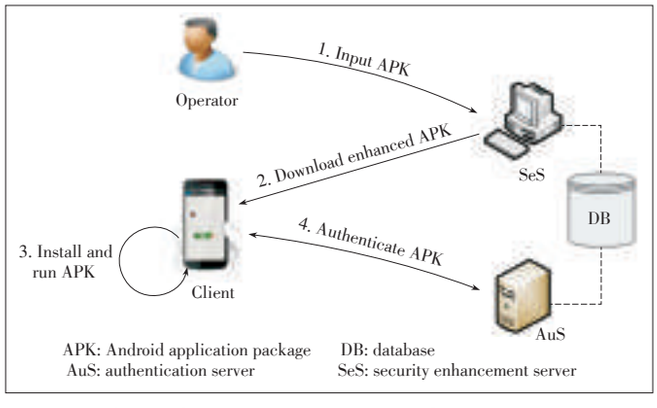
The operator of SeSoa submits the original unprotected APK to SeS. SeS completes all the functions of security enhancement: 1) using the encryption technique to encrypt the dex code in the application, so that the plaintext code no longer appears on the client, which ensures that the code is protected from static analysis; 2) adding the safety function code to make sure that the application can be protected from dynamic debugging; 3) adding the code supporting authentication function so that the client could run the particular APK only after the completion of the authentication by the AuS.

2) Download enhanced APK

The enhanced APK can be downloaded by every Android user to the Android mobile terminals.

3) Install and run APK

The users install and run the enhanced APK.



▲Figure 1. System framework and operational flows.

4)Authenticate APK

The enhanced APK needs to complete the online authentication function when it is run for the first time. Only after it passes the authentication by the AuS can the enhanced APK get the security parameters, which allow it to continue running and move into the APK original function module.

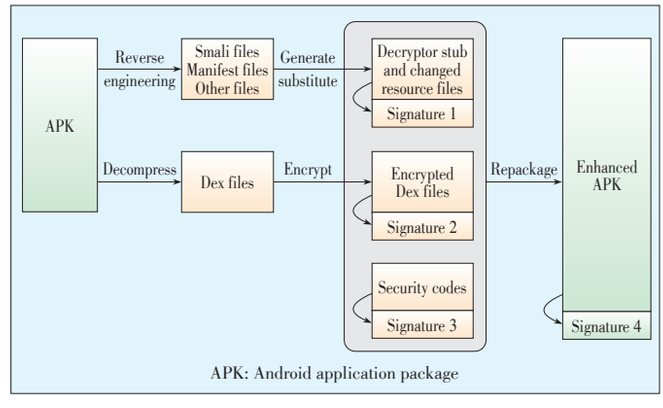
In addition, the SeS and AuS are logical concepts, which can be separated or implemented on one equipment. However, they can share the same Database to reduce the complication of management for data consistency.

3.2 Security Enhancement Process

The security enhancement process is implemented on SeS (**Fig. 2**). At the first step, the original APK file needs two pretreatment functions. One is to gain information about the APK and its implementation details, including the Smali and Manifest files using reverse engineering tools such as APKTool [10]. The other is to decompress the APK to get the .dex file. After the pretreatment process, the files are ready to be used in the following steps.

The two components of the encrypted application and a decryptor stub are generated at the second step. Unlike Proguard and Dexguard using obfuscation to make the code harder to be understood, SeSoa encrypts the .dex file and decrypts it before it is loaded into memory. In this way, the .dex file will not be stored as the plain text in the file system, which means that the others must decrypt it first if they want to obtain the code. To make the encrypted .dex file run normally in the Android system, the decryptor stub has to be implemented to fulfill the fetch of the encrypted application into memory, decrypt the application and yield the original .dex file, which can be loaded into the DVM and executed. Because the decryptor stub should be run first, so we have to change the manifest file to ensure that the application can be run normally. These changes include replacement of the component names, modification of the primary activities property, addition of some new services and activity components, etc.

In order to attack such a security enhancement scheme, a reverse engineering tool has to gain access to the decrypted .dex



▲Figure 2. The security enhancement process of SeSoa.

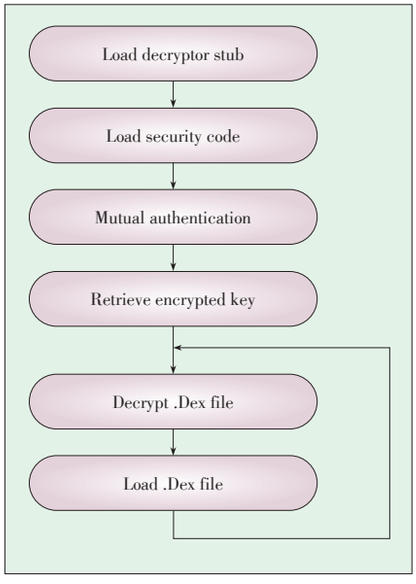
file. This can only be done after the .dex file is successfully decrypted and loaded into the memory. To protect from such kind of threats, some other security mechanisms such as anti-dynamic debug function are needed. More importantly, the client also has no information about the source of the APK through this method, so the authentication mechanisms should be applied to make sure that the APK is from the authorized party. All these security mechanisms besides encryption/decryption are included in the security code module in Fig. 2.

At the third step, the signature of the resource files (signature 1), the signature of the original .dex files (signature 2) and the signature of security codes (signature 3) are generated separately according to different parts of data. These signatures are used to protect the APK from being tampered.

Finally, all the files in the application are repackaged, and need to be resigned as any normal applications should do, as the Android system requires that developers must sign applications with their private key/certificate unless the application cannot be installed on user devices. Furthermore, because of the online authentication function, the data (application ID, secret K, CKSUM, Addr) are sent and stored in the database on AuS for further use.

3.3 Application Load Process

After APK is downloaded and installed in the Android client side, the decryptor stub program is loaded and run at once. The decryptor stub calls the online authentication module to communicate with AuS, and only those applications that pass the authentication verification can get the correct address of the required Dex. Then the Key used for encrypting .Dex is retrieved and used to decrypt .Dex file, and the successfully decrypted .Dex file is the original .Dex file of the application, which can be loaded into DVM and executed. **Fig. 3** shows this application load process.



◀**Figure 3. Application load process.**

In fact, the process is not as easy as what is described above because we have to make sure that the whole process about decryption is secure and other applications cannot get the plain text of .dex file through the process. DVM also has limited instruction set, so there is no direct way to access the bytecode with an instruction, which makes it impossible to execute the bytecode stream during the running time. To circumvent this restriction, "Java Native Interface" (JNI) of the DVM is used, JNI is intended to allow execution of native code [11]. The mutual authentication process is another complicated process, the detailed authentication protocol and processes in SeS and Android client are described in the following section.

3.4 Online Authentication Process

The online authentication process is used to achieve the mutual authentication between the Android client and authentication sever when an app is installed and run for the first time. Considering the performance of the implementation on Android side, only the symmetric cryptography is used in the scheme. **Table 1** lists the notations used in this section.

Fig. 4 shows mutual authentication procedure of the proposed Android application security enhancement scheme. This procedure is triggered by the application at the Android side after the decryptor stub starts and finishes generating the check sum of the application (CKSUM), as well as retrieving the K from the application.

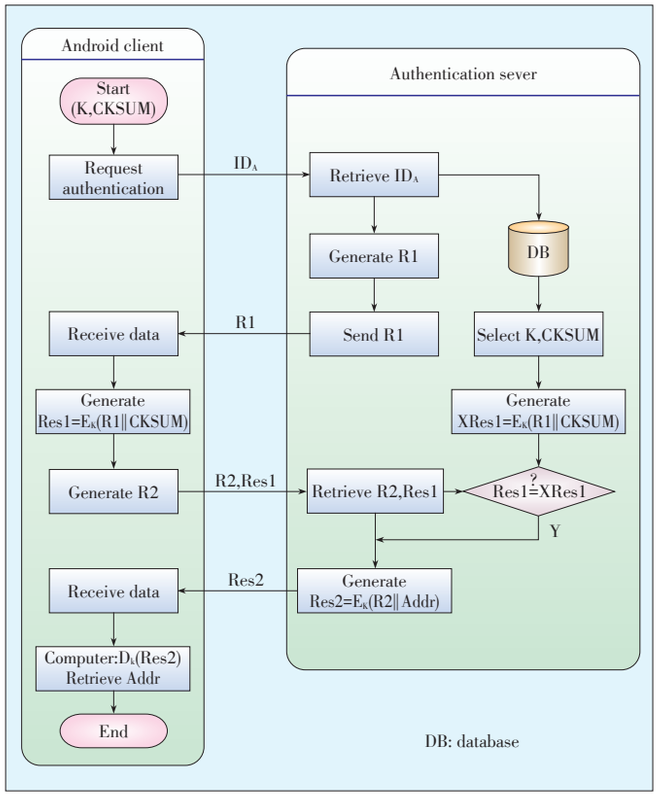
1) The application on Android client side sends an authorization request message to the AuS, which contains its own identifier ID_A;
2) AuS receives the message and retrieves ID_A from the authentication request message, then it can use ID_A to select K and CKSUM from database, which are stored in the database by SeS. At the same time, the SeS generates a random number R1, and sends it to the Android client. After that, it calculates XRes1 using such symmetric algorithm as DES or AES with R1 and CKSUM as the input parameters and K as the key. XRes1 will be stored and used later.
3) When the client application receives R1, it can calculate Res1 with the same algorithm running on AuS, using K and

▼**Table 1. Notations**

Notions	Description
K	Symmetric key
CKSUM	Integrity check value
ID_A	Identification of application
Addr	Start address of main Dex program
R1/2	Random number
Res1/2	Response value
XRes1	Expect response value
E_K	Encryption using K
D_K	Decryption using K

SeSoa: Security Enhancement System with Online Authentication for Android APK

DONG Zhenjiang, WANG Wei, LI Hui, ZHANG Yateng, ZHANG Hongrui, and ZHAO Hanyu



▲Figure 4. Authentication scheme.

CKSUM as the algorithm's inputs. Then the app generates random number R2, and sends R2 and Res1 back to the SeS.

4) After SeS retrieves R2 and Res1, Res1 is compared with XRes1. If they are equal, it is shown that the application in Android is enhanced by SeS and the application has not been modified. If they are not equal, the app cannot get the K and correct CKSUM, and AuS will send error number to the client and stop the process. Secondly, SeS generates Res2 using encryption algorithm, Res2 equals E_k(R2||Addr). Finally, SeS sends Res2 back to the client.

5) At the Android side, Res2 can be retrieved from the message and decrypted to R2'||Addr' by the decryption algorithm using the same key K. Then R2' and R2 is compared, if they are equal, it means that this message is sent by the real SeS, and the application can regard addr' as the correct security parameter, which can be some code in the initial process or some key data needed by the application; otherwise, the application can react as what its security strategy allows.

4 Experiments and Evaluations

In this section, the test results of SeSoa on compatibility and performance are presented, which indicates that SeSoa is usable. The security of SeSoa is also analyzed to ensure that it can also meet the requirement for security.

4.1 Compatibility Analysis

Besides the security goals, SeSoa should also possess the capability of enhancing arbitrary application automatically. To test this feature, 50 applications in Android were downloaded and applied in our test. These applications were randomly selected from Android App Store, loaded and run in the 100 types of mobile phones with Android 4.0–4.4 platforms. For each application, we first evaluated whether it could be enhanced by SeSoa. If the answer is positive, we evaluated whether it could be loaded and run in the 100 types of smartphones. Experimental results are shown in **Table 2**.

The success rate for SeSoa to enhance the random applications is 82%, which is acceptable, although not very high. However, as the complexity of the applications increases, the success rate of the reinforcement system will decline. Therefore, it is necessary to further optimize the program.

4.2 Performance Analysis

In order to test whether SeSoa seriously affects the performance of the original application, ten test applications were randomly selected out of the above mentioned 41 apps that were successfully run on our 100 test mobile platforms. Because the running time varies on different platforms, the experiments presented in this section were conducted using Samsung Galaxy S4 smartphone, equipped Exynos 5410 dual quad-core processor and 2 GB RAM.

Table 3 shows the test results of performance. It can be seen that after being enhanced by the SeSoa, the installation package size of the applications increases about 10%, the average start-up time is increased by about 20%. Since the start-up time is generally less than 1 sec, this will not make users feel different compared with the start-up time of original APPs.

4.3 Security Analysis

1) Anti-tamper

We use the multiple signature mechanism to prevent the application from being tampered. In the security enhancement process, three signatures are generated at the third step. Therefore, at the client side, the signature of the resource files, the signature of the original .dex files and the signature of security code are verified during the application's start-up process, be-

▼Table 2. Test results for compatibility

App size	App numbers	Enhanced APP numbers	Success numbers for loading	Success numbers for runing	Platform compatibility
<1 MB	11	11	11	11	99.3%
1 MB–5 MB	25	23	23	22	98.6%
>5MB	14	10	9	8	96.4%
Sum	50	44	43	41	98.1%
Success rate	-	88%	86%	82%	-

▼ **Table 3. Test results for performance**

App no.	Original app size (KB)	Enhanced app size (KB)	Size change rate (%)	App startup time (ms)	Enhanced app startup time (ms)	Startup time increase rate (%)
1	560	581	3.8	284	365	28.5
2	740	784	5.9	415	467	12.5
3	813	897	10.3	396	452	14.1
4	1123	1260	12.2	675	813	20.4
5	1457	1634	12.1	561	697	24.2
6	2360	2523	6.9	741	879	18.6
7	2709	2942	8.6	759	896	18.1
8	3234	3558	10.0	737	834	13.2
9	4921	5489	11.5	891	1019	14.4
10	8913	9974	11.9	919	1131	23.1

fore the decryptor stub begins to decrypt .dex files. If any of the three verifications fail, the application will stop running. In this way, any tamper of the application will be detected.

The other anti-tamper technique we used is online authentication. Only the application that has passed the verification of AuS can obtain the secure parameters to run the core code in the APK. Any modification of the APK will lead to the failure of generating the correct CKSUM and secret K, so the APK on the Android client can hardly calculate the correct response of R1, which can be regarded as the challenge of AuS in a challenge-response protocol.

2) Preventing dynamic debugging

In our solution, a monitoring process ring is used to monitor whether there are some system calls for Ptrace, which is used by debuggers and other code - analysis tools to analysis the code.

The mainstream debugging tools use Ptrace system calls in Linux system; if a process is calling Ptrace, there must be someone using such kind of tools to debug and that the application should be stopped. This process for monitoring the call of Ptrace is used by most application enhancement systems, but the risk for such a method is that attackers can stop the monitoring process and then debug the application again. To avoid this weakness, we designed the monitoring process ring mechanism.

Our monitoring process ring scheme has three processes: the parent process (the main process to be protected), the child process (generated by the parent process) and the grandchild process (generated by the child process). These processes compose a process ring and monitor each other in two levels. The first level of monitoring is to ensure that the process of the ring is not damaged, which means that these three processes will not be killed. This can be achieved by monitoring the pipeline. The father process and child process listen anonymous pipes between them, so do the child process and the grandchild process. The father process monitors the other processes by listening to the named pipe. Once a process is killed, the other end of the pipe will be aware of, and the related process will enter its protection module to end the parent process. The second level of monitoring helps ensure that the process ring not be debugged by monitoring the status of the other two processes, using two threads created by each process, the parent process will be ended in case that the process finds its child or grandchild process' status changed to suspended, which indicates that the process is debugged by the other process.

Our solution also monitors the communication of Java Debug Wire Protocol (JDWP) to avoid the application from dynamic debugging on the java layer. The strategy is to find out all Java debug tools using JDWP in their socket communications and to judge what kind of debug tool is connected to. If the debug tool belongs to Andbug, which is usually used by a debugger to reverse-engineer applications for the DVM on Android platform, then the process will be killed at once.

3) Preventing decompilation

The main idea for preventing decompilation is encryption. The .dex files are encrypted with symmetric cryptographic algorithm. The key used in the algorithm is hidden among the codes of the application. In addition, we implement the White Box Cryptography of AES and SMS4, which is used to encrypt the symmetric cryptographic keys so that the encryption algorithm can be run in an unsafe environment of Android system.

5 Conclusions

There are great demands for code protection to prevent Android applications from being reversed and tampered, because the source codes of the Android applications can be more easily recovered by the existing reverse engineering methods. In this paper, we propose a security enhancement system with online authentication for Android APK called SeSoa, in which multiple security mechanisms such as encryption, mutual authentication and monitoring process ring are used to protect Andriodapplications from tamper, dynamic debugging and decompile. To achieve the good balance between security and usability, the mutual authentication is only proceeded when an enhanced Android application is installed and run at the first time.

References

[1] Android Open Source Project. (2015, Oct.). *Android security overview* [online]. Available: https://source. android.com/security/index.html
[2] AppBrain. (2015, Nov.). *Number of android applications* [online]. Available: http://www.appbrain.com/stats/number- of-android-apps
[3] V. Oorschot, and C. Paul, "Revisiting software protection," in *Information Security*. Germany: Springer Berlin Heidelberg, 2003, pp. 1–13.
[4] Sourceforge. (2015, Oct.). *ProGuard* [online]. Available: http://proguard.sourceforge.net
[5] C. Coakley, J. Freeman, and R. Dick. (2005, Feb. 4). *Next-generation protection against reverse engineering* [Online]. Available: http://www.anacapasciences.com/

publications/protecting_software2005.02.09.pdf
[6] C. Kruegel, W. Robertson, F. Valeur, and G. Vigna, "Static disassembly of obfuscated binaries," in *USENIX security Symposium*, San Diego, USA, 2004, pp. 18–18.
[7] DexGuard. (2015, Oct.). *DexGuard, premium security software for android applications* [online]. Available: http://www.saikoa.com/dexguard
[8] Allotori. (2015, Oct.). *Allatori java obfuscator* [online]. Available: http://www.allatori.com
[9] Y. Piao, J. H. Jung, and J. H. Yi, "Server-based code obfuscation scheme for APK tamper detection," *Security and Communication Networks*, vol. 9, no. 6, pp. 457–467, 2014. doi: 10.1002/sec.936.
[10] APKTool. (2015, Oct.). *A tool for reverse engineering android apk files* [online]. Available: http://ibotpeaches.github.io/Apktool
[11] P. Schulz. (2012, Jun. 7). *Code protection in android* [Online]. Available: http://net.cs.uni-bonn.de/fileadmin/user_upload/plohmann/2012-Schulz-Code_Protection_in_Android.pdf

Manuscript received: 2016-01-10

Biographies

DONG Zhenjiang (dong.zhenjiang@zte.com.cn) received his MS degree from Harbin Institute of Technology, China. He is the leader of the Business Expert Team of Expert Committee for Strategy and Technology of ZTE Corporation and the deputy president of Cloud Computing and IT Research Institute of ZTE Corporation. His research interests include cloud computing, big data, new media, and mobile internet. He has led more than ten funded programs and published a monograph and more than ten academic papers.

WANG Wei (wang.wei8@zte.com.cn) received her BS degree from Nanjing University of Aeronautics and Astronautics, China. She is an engineer and project manager in the field of mobile internet at Cloud Computing and IT Research Institute of ZTE Corporation. Her research interests include new mobile internet services and applications, PaaS, and terminal application development. She has authored five academic papers.

LI Hui (lihuill@bupt.edu.cn) received her PhD in cryptography in 2005 from Beijing University of Posts and Telecommunications (BUPT), China. From July 2005, she has been working for School of Computer Science at BUPT as a lecturer and associate professor. Her research interests are cryptography and its applications, information security and wireless communication security.

ZHANG Yateng (526551337@qq.com) is a graduate student in School of Computer Science at BUPT. His research interests include smart phone security, application of cryptographic algorithms, and implementation of white-box encryption algorithm on mobile platform.

ZHANG Hongrui (zhanghongrui@bupt.edu.cn) is a graduate student in School of Computer Science at BUPT. He is conducting research on information security and software protection.

ZHAO Hanyu (hyzhao1990@163.com) is a graduate student in School of Computer Science at BUPT. He is conducting research on software protection in smartphone.

DOI: 10.3969/j. issn. 1673-5188. 2016. S0. 007
http://www.cnki.net/kcms/detail/34.1294.TN.20160621.1739.002.html, published online June 21, 2016

Screen Content Coding in HEVC and Beyond

LIN Tao, ZHAO Liping, and ZHOU Kailun
(Tongji University, Shanghai 200092, China)

Abstract

Screen content is video or picture captured from a computer screen typically by reading frame buffers or recording digital display output signals of a computer graphics device. Screen content is an extremely comprehensive and diverse class of content and includes traditional photosensor captured pictures as a small subset. Furthermore, screen content has many unique characteristics not seen in traditional content. By exploring these unique characteristics, new coding techniques can significantly improve coding performance of screen content. Today, more than ever, screen content coding (SCC) is becoming increasingly important due to the rapid growth of a variety of networked computers, clients, and devices based applications such as cloud computing and Wi-Fi display. SCC is the ultimate and most efficient way to solve the data transferring bottleneck problem in these applications. The solution is to transfer screen pixel data between these computers, clients, and devices. This paper provides an overview of the background, application areas, requirements, technical features, performance, and standardization work of SCC.

Keywords

HEVC; AVS; Screen Content Coding; String Matching; Video Coding

1 Introduction

The screen content coding (SCC) standard [1] for high efficiency video coding (HEVC) is an international standard specially developed for screen content. It indicates the start of a new chapter in video coding research and standardization. On one hand, SCC is required by many traditional applications and an ever-increasing number of new and emerging applications as well [2]–[4]. On the other hand, screen content is very different from traditional content, thus different coding tools are needed. Furthermore, screen content is an extremely comprehensive and diverse class of content and includes traditional photosensor (e.g. CMOS or CCD sensor) captured pictures as a small subset. As a result, SCC is becoming a very active field to attract considerable attention from both academia and industry [1]–[35], and is expect to play a major role in advancing both researches and applications of video coding technology.

The Audio Video Coding Standard (AVS) Workgroup of China is also working on SCC standard, which is expected to become a national standard in China and an IEEE standard by the second half of 2016. Since SCC has much more application areas, market sectors, and customers of different requirements to serve than traditional video coding, multiple standards are needed and benefit each other to grow the market size. There are also many SCC application areas and market sectors where proprietary solutions are also acceptable.

This paper discusses the background and current status of SCC and its standardization work in HEVC and AVS. The rest of the paper is organized as follows. In section 2, application areas and requirements of SCC is presented. Section 3 describes characteristics of screen content. Section 4 is devoted to technical description and standardization of three major dedicated SCC techniques and their relation. Section 5 reports coding performance comparisons of the three SCC techniques. Finally, Section 6 concludes the paper and also presents some future work of SCC.

This work was supported in part by National Natural Science Foundation of China under Grant No. 61201226 and 61271096, and Specialized Research Fund for the Doctoral Program under Grant No. 20130072110054.

2 Application Areas and Requirements of SCC

Almost all applications of SCC have one thing in common: display units are connected to information processing resources, including the central processing unit (CPU), graphics processing unit (GPU), and storage space, through networks.

Application areas of traditional video coding are mostly related to TV broadcasting, video content delivery or streaming, and video surveillance. However, SCC opens a huge and new application area of video coding: cloud computing platform, where CPUs, GPUs, and main storages devices are all located in a place called cloud and shared by multiple user (client) devices that are connected to the cloud through networks. As shown in **Fig. 1**, the cloud can be as big as a datacenter with thousands or tens of thousands of servers or as small as a single computer with one multi-core-CPU/GPU combo or even a smart phone. Virtual network computing (VNC), remote desktop, virtual desktop infrastructure (VDI), PC over IP (PCoIP),

▲**Figure 1. Application scenario of SCC: Cloud/cloudlet and cloud-mobile computing platform.**

ultra-thin client, and zero-client are a few examples of SCC based cloud computing platform implementation. SCC based implementation has the highest graphics performance among all implementations of the cloud computing platform [2]–[4]. SCC can reduce screen pixel data bit-rate to a level that widely deployed networks can support even for screen resolution of 2560x1600 or higher at 60 Hz screen refresh rate, thus enables cloud based computing and information processing to become a mainstream model not only used by professionals but also by average people in their daily life. The daily activities that often need to handle typical screen content include web browsing, document sharing in video conferencing, remote desktop sharing and collaboration, office document editing, engineering drawing, hardware design engineering, software programming, map navigating and address direction searching, and many more. Therefore, the market of SCC based cloud computing and its variations are expected to grow exponentially and its market is becoming much bigger than traditional video coding market.

Besides cloud computing platforms, SCC has at least the following application areas:

- Cloudlet computing, a variation of cloud computing, where the cloud is a small one (cloudlet) or is split into a few small cloudlets. A client device can become a cloudlet.
- Cloud-mobile computing, a variation of cloud computing where the client devices are mobile devices such as smart phones, tablets, or notebooks
- Cloud gaming
- Wireless display, for example, Wi-Fi display, where Wi-Fi connection is used to replace a video cable that attaches a display unit such as a monitor or a TV set to a PC, a notebook, a tablet, a smart phone, a set-top-box, and so on
- Screen or desktop sharing and collaboration, where multiple users at different locations view the same desktop screen
- Video conferencing with document sharing
- Remote teaching
- Display wall
- Multi-screen display for many viewers
- Digital operating room (DiOR) or OR-over-IP.

From SCC coding performance and coding quality point of view, SCC application areas and markets can be divided into the following two major segments, which have different requirements.

1) High and ultra-high video quality segment

This segment includes cloud/cloudlet/cloudlet- mobile computing platforms, enterprise IT cloud platforms, VDI, remote desktops, PCoIP, ultra-thin clients, zero-clients, cloud gaming, and more. One distinct feature of this segment is that the screen is usually viewed by one viewer at a viewing distance less than one meter. This viewing model is the same as what traditional computer users normally do. Users may include professionals, and no visual loss of screen content picture can be tolerated. Due to this feature, lossless coding or visually lossless coding with high and ultra-high picture quality is an absolute requirement in this segment. The video color format requirement for this segment is RGB or YUV 4:4:4. Another distinct feature of this segment is that human-computer interaction (HCI) is involved, and both encoding and decoding of screen content are part of the HCI process [2]. The total round-trip time from a keyboard input or a mouse click to task processing on the cloud, screen content rendering on the cloud, screen content encoding on the cloud, and finally screen content decoding on the client device should be within a limit that users can accept. Thus, the encoding and decoding time and latency are very important to get overall crisp system response time (SRT) for uncompromised excellence of HCI experiences. The total encoding and decoding latency requirement is typically 30 milliseconds or less. This is less than one frame period in 30 frames per second coding configuration. Therefore, in this segment, contrast to traditional video coding applications, peak intra-picture (all-intra) coding performance is far more critical than random-access, low-delay-P, and low-delay-B coding performance. In this segment, the highest mainstream screen resolution today and in near future is probably 2560x1600 pixels. At 60 frames per second screen refresh rate and 24 bits per pixel color precision, the raw screen pixel data bit-rate is 5626 mega bit per second (mbps). Today, advanced widely deployed networks infrastructure can probably provide sustainable bandwidth of up to 20 mbps. Therefore, the basic compression ratio requirement is close to 300:1. The compression ratio at ultra-high visually lossless picture quality is certainly very challenging, especially in all-intra coding configuration.

2) Middle and low video quality segment

This segment includes Wi-Fi display, display wall, external second display of mobile devices, multi-screen display for many viewers, video conference with document sharing, remote teaching, and more. One distinct feature of this segment is that the screen content is usually viewed by more than one viewer at a viewing distance more than one meter. This viewing model

is not much different from traditional TV viewing model. Due to this feature, lossy coding with middle (or low in some cases) picture quality is acceptable in this segment. The video color format requirement for this segment is RGB or YUV 4:4:4 or YUV 4:2:0. Another distinct feature of this segment is that human-computer interaction (HCI) is usually not involved. So, the encoding and decoding latency is not as important as in the first segment. All-intra coding performance is also not as important as in the first segment. In this segment, the highest screen resolution today and in near future is probably 4096x2160 pixels. At 60 frames per second screen refresh rate and 24 bits per pixel color precision, the raw screen pixel data bit-rate is 12,150 mbps. Today, advanced widely deployed networks infrastructure can probably provide sustainable bandwidth of up to 20 mbps. Therefore, the basic compression ratio requirement is 600:1. The compression ratio of 600:1 is certainly very challenging even at middle picture quality.

As a result, SCC requirements for compression ratio and picture quality are very challenging. Traditional coding techniques cannot meet the requirements, and new coding techniques are absolutely needed.

3 Characteristics of Screen Content

One of the most important characteristics of screen content is its diversity and comprehensiveness.

Screen content is video or picture captured from a computer screen typically by either reading frame buffers or recording digital display output signals of a computer graphics device. Computer screen content is extremely diverse and comprehensive due to the diversity and comprehensiveness of materials, data, information, and their visual bitmap representations that computers need to handle, render and display,. The diversity and comprehensiveness can be seen from at least the following three aspects:

1) The number of distinct colors in a region (e.g. a block). The number can range from one, i.e. the entire region has only one color, to maximum, which is equal to the number of pixels in the region;
2) Degree of pattern matchability. A matching (either exact matching defined as having no difference or approximate matching defined as having difference within a predetermined limit) pattern set is a set of two or more patterns that have both matching shapes and matching value of pixels. Pattern matchability is the state of existence of matching pattern sets. The degree of pattern matchability can be measured by at lease the following metrics:
   - The size (the number of elements) of a set of matching patterns in a predetermined range usually called searching range in an encoder or a reference range in a decoder. Big size means high degree of pattern matchability.
   - The number of matching pattern sets in a predetermined range. The number equal to 0 means the lowest degree of pattern matchability. The degree of pattern matchability is generally related to both of the number of matching pattern sets and the average size of all matching pattern sets. In general, big number of matching pattern sets or big average size of all matching pattern sets means high degree of pattern matchability.
   - The average distance between elements of a matching pattern set. Short distance usually means high degree of pattern matchability and that most elements are located closely.

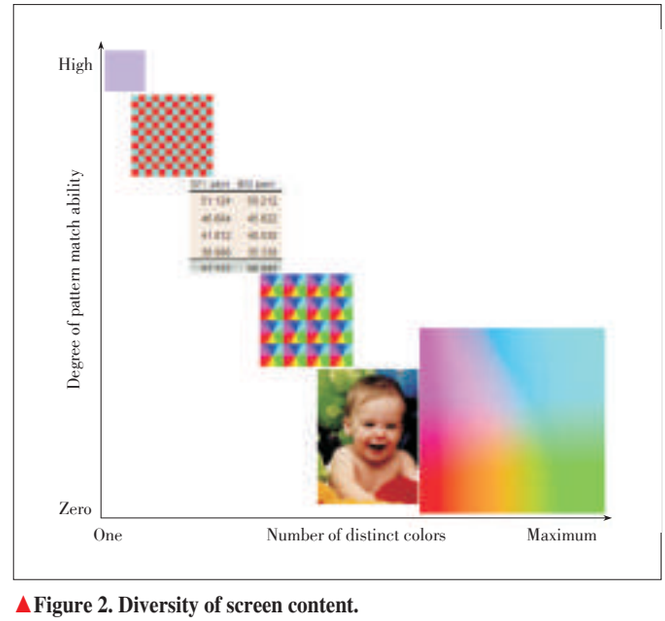
3) Shape and color of matching patterns

The shape and color of a matching pattern set can be arbitrary. Hence, the number of possible different shapes and colors is huge. In fact, matching patterns in screen content have a virtually unlimited number and variety of different shapes and colors. For example, the shapes range from simple ones such as squares, rectangles, triangles, circles, polygons, crescents, diamonds of different sizes to complex ones such as all kinds of mathematic curves, geometric shapes, and fonts of different typeface, size, weight, slope, width, and special effect.

In **Fig. 2**, six screen content examples illustrate the screen content diversity in terms of the number of distinct color.

As shown in Fig. 2, from top left to bottom right, the six examples are:

1) Single color square. The number of distinct color is one. The degree of pattern matchability is the highest.
2) Two color checkers. The number of distinct color is two. The degree of pattern matchability is very high.
3) Spreadsheet cells. The number of distinct color is about twenty. The degree of pattern matchability is high.
4) A color space diagram repeated 16 times. The number of distinct color is big. The degree of pattern matchability is medium.



▲**Figure 2. Diversity of screen content.**

5) A photosensor (camera) captured photo, i.e. a natural picture. The number of distinct color is big. The degree of pattern matchability is very low.
6) A color space diagram. The number of distinct color reaches the maximum. The degree of pattern matchability is zero.

It should be noted that screen content includes the traditional photosensor captured natural content as a small subset of screen content. In fact, this special subset generally features a very large number of distinct colors and almost zero pattern matchability, as the fifth example in Fig. 2.

Screen content also includes sophisticated light-shaded and texture-mapped photorealistic scenes generated by computers. Virtual reality, 3D computer animation with lighting and shading, 3D computer graphics with lighting and shading, and computer games are examples of photorealistic screen content. From the word "photorealistic", it can be easily seen that computer generated photorealistic screen content has almost the same properties as the traditional photosensor captured natural content and share the same features like relatively smooth edges and complicated textures. In particular, computer generated photorealistic screen content also features a very large number of distinct colors and almost zero pattern matchability.

Besides the general characteristics of diversity and comprehensiveness, typical screen content has at least the following three specific characteristics, which traditional natural content usually does not have.

1) Typical computer screens seen in common everyday applications are often rich in girds, window frames, window panes, table cells, slide-bars, toolbars, line charts, and so on. They feature very sharp edges, uncomplicated shapes, and thin lines with few colors, even one-pixel-wide single-color lines. Therefore, for typical screen content, the number of distinct colors is low.
2) Sharp and clear bitmap structures, especially small ones, such as alphanumeric characters, Asian characters, icons, buttons, graphs, charts and tables are often seen in typical computer screens. Thus, there are usually many similar or identical patterns in typical screen content. For examples, all texts are composed of a very limited number of characters, and all characters themselves are composed of a significantly further limited number of basic strokes. Therefore, typical screen content has high degree of pattern matchability.
3) Splitting and merging of matching patterns. A pair of matching patterns (A, B) with a pair distance d(A, B) may be split into two or more pairs of small matching patterns (A_1, C_1), (A_2, C_2), ⋯, where pattern A is split into two or more small patterns A_1, A_2, ⋯, and each pair of the small matching patterns has a pair distance shorter than d(A, B). On the other hand, if pattern A is split into two or more small patterns A_1, A_2, ⋯, two or more pairs of matching patterns (A_1, C_1), (A_2, C_2), ⋯ may be merged into a big pair of matching pattern (A, B), whose pair distance is longer than the pair distance of each of the small matching pattern pairs (A_1, C_1), (A_2, C_2), ⋯ Note that in splitting and merging, it is not necessary for patterns C_1, C_2 ⋯ to be related to pattern B. The splitting and merging of matching patterns mean that for a piece (or block) of pixels in typical screen content, matching relation is not unique, but has multiple options available. Different options have different number of pairs and different pair distances. Pattern splitting and merging based multiple matching relation is an important characteristic which needs to be fully explored in SCC.

The first two specific characteristics are strongly related. Actually, lower number of distinct colors usually (but not always) means higher degree of pattern matchability and vice versa. On one extreme, one distinct color results in the highest degree of pattern matchability; at the other extreme, the maximum number of distinct colors results in zero degree of pattern matchability.

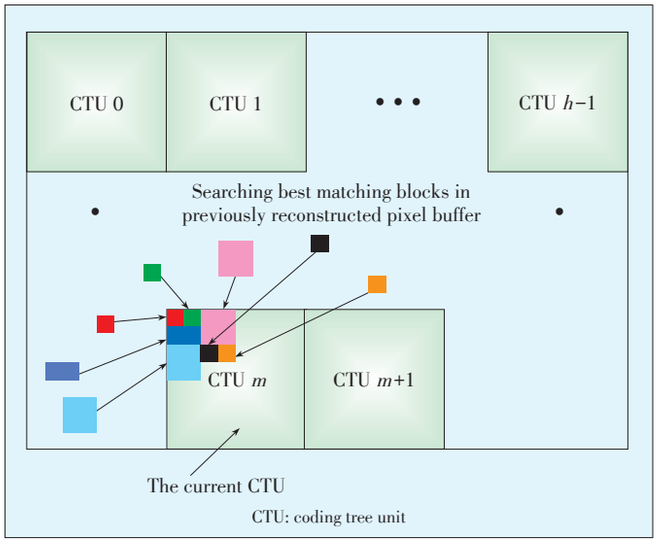
Because traditional block-matching and transform based hybrid coding technique does not take advantage of any special characteristics of screen content, dedicated SCC techniques have tremendous potential to improve coding efficiency of screen content significantly by exploring the special characteristics of screen content. Since almost all major characteristics of screen content are related to pattern matching, three major SCC techniques are all pattern matching based techniques: 1) intra picture block matching technique, also known as intra block copy (IBC) or intra motion compensation (IMC)technique; 2) intra coding unit (CU) pixel index string matching technique, also known as palette (PLT) technique; and 3) pseudo 2D string matching (P2SM or P2M) technique.

4 Technical Description and Standardization of Three SCC techniques

IBC [5]–[8] is a straightforward extension of conventional inter-prediction to intra picture coding with a few simplifications. The main simplification is to remove pixel interpolation and do only whole pixel prediction. In IBC (**Fig. 3**), when encoding a prediction unit (PU), the encoder searches an optimal matching block as a reference matching pattern in a pre-determined search window (reference buffer), which is usually a previously reconstructed pixel buffer. Reference matching patterns have the same shapes and sizes as PUs such as 4x8, 8x4, 8x8, 16x16, 32x32, 64x64 pixels. The search window has a predetermined size which varies from a few CUs to the full frame. The encoding result is a motion vector (MV) and a residual block.

In IBC decoding, the decoder parses the bitstream to obtain a MV. The decoder uses the MV to locate a reference matching pattern in the previously reconstructed reference pixel buffer. The decoder then uses the values of the reference matching pattern as the predictor of the current PU being decoded.

IBC is efficient to code matching patterns of a few fixed siz-

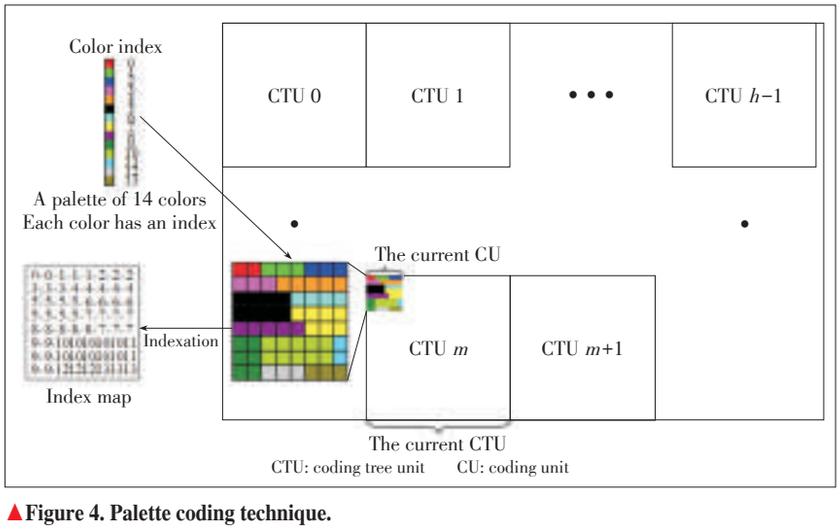


▲Figure 3. Intra block copy coding technique.

es with rectangle or square shapes in a picture, but is not flexible enough to code matching patterns of different sizes from a few pixels to a few thousands of pixels with a variety of shapes.

IBC is adopted into the HEVC SCC draft by unification with conventional inter-prediction, i.e. specifying the current picture itself as a reference picture.

As shown in **Fig. 4**, when encoding a CU in PLT [9]–[11], the encoder first performs color quantization on the CU to obtain a few representative colors and puts the representative colors into a palette. Each color in the palette has an index. When the number of representative colors exceeds a limit, the last index is reserved to represent all extra colors beyond the limit. The extra colors are named escape colors. All pixels in the CU are converted into indices to build an index map. The index map is further coded by either left-string-matching or above-string-matching. The escape colors are quantized and coded into the bitstream.



▲Figure 4. Palette coding technique.

All indices in an index map are coded string by string using two types of string matching. The first type of string matching is left-matching. The first string (0 0), second string (1 1 1), and third string (2 2 2) in the index map of Fig. 4 are examples of left-matching. In a left-matching string, all indices are identical. The second type of string matching is above-matching. In the index map of Fig. 4, string (5 5 5 5) in the 4th row, string (7 7 7) in the 5th row, string (9 9 10 10 10 10 10 11) in the 7th row, and string (9 9) in the 8th row are examples of above-matching. It is obvious that in left-matching, the reference matching string (pattern) overlaps the current string being coded, while in above-matching, the reference matching string (pattern) is the string above the current string being coded. A left-matching string has three coding parameters: string type, index, and length. An above-matching string has two coding parameters: string type and length.

For each CU, the PLT encoding results are a palette, an index map coded by two types of string matching, and quantized escape colors. The encoding results are explicitly or implicitly put into the video bitstream after entropy coding.

In PLT decoding, the decoder parses the bitstream and performs other decoding steps to obtain the palette, the index map, and the escape colors, from which the decoder can complete the decoding process to reconstruct all pixels of the CU.

The palette coding technique can code matching patterns inside a CU using two types of intra-CU pixel-index string matching, but it cannot exploit non-local matching patterns outside of a CU.

PLT is adopted into the HEVC SCC draft as a CU level coding mode named palette mode.

IBC can only code matching patterns of a few fixed sizes with rectangle or square shapes efficiently. PLT can only code matching patterns completely inside a CU efficiently. However, typical screen content shows significant diversity in terms of the shape and size of matching patterns and the distance of a matching pattern pair. Therefore, IBC and PLT only partially explore the special characteristics of screen content.

P2SM has its origin in Lempel-Ziv (LZ) algorithm [12], but is more sophisticated than the original LZ algorithm. In P2SM, two reference buffers are used. One is primary reference buffer (PRB) which is typically a part of the traditional reconstructed picture buffer to provide reference string pixels for the current pixels being coded. The other is secondary reference buffer (SRB) which is a dynamically updated look-up table (LUT) storing a few of recently and frequently referenced pixels for repetitive reference by the current pixels being coded. When encoding a CU, for any starting pixel being coded, searching of optimal matching string with a variable length is performed in both PRB and

SRB. As a result of the searching, either a PRB string or an SRB string is selected as a reference matching pattern on a string-by-string basis. For a PRB string, an offset and a length are coded into the bitstream. For an SRB string which is really an SRB pixel color duplicated many times, an SRB address and a duplication count are coded into the bitstream. If no reference string of at least one pixel is found in PRB or SRB, the starting pixel is coded directly into the bitstream as an unmatched pixel. Thus, a CU coded by P2SM has three matching types: Match_PRB, Match_SRB, and Match_NONE. A letter S coded by P2SM is shown in **Fig. 5**. The size of the current CU is 8x8. The following is five examples of PRB strings or SRB strings (Fig. 5).

The 1st string marked with red "1" is a 9-pixel PRB string. The reference matching string is in PRB with offset (9, 3).

The 2nd string marked with green "2" is a 4-pixel SRB string. The reference matching string consists of the 1st SRB pixel color duplicated four times.

The 3rd string marked with red "3" is a 4-pixel PRB string. The reference matching string is in PRB with offset (0, 3).

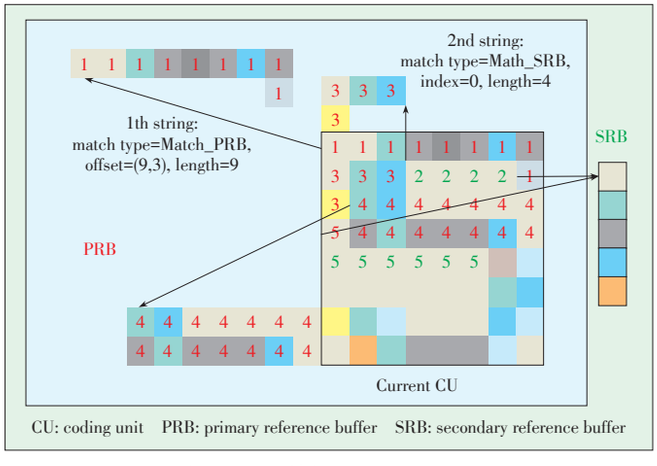
The 4th string marked with red "4" is a 14-pixel PRB string. The reference matching string is in PRB with offset (8, -4).

The 5th string marked with green "5" is a 7-pixel SRB string. The reference matching string also consists of the 1st SRB pixel color duplicated seven times.

In P2SM decoding, the decoder parses the bitstream and performs other decoding steps to obtain the matching type, (offset, length) or (SRB address, length) or unmatched pixel, from which the decoder can complete the decoding process to reconstruct all pixels of the CU.

P2SM is adopted into the initial working draft of AVS screen mixed content coding extension as a CU level coding mode in March 2016.

It is obvious that IBC and PLT are two special cases of P2SM. In fact, IBC is a P2SM special case that restricts a PU to have only one reference matching string. PLT is also a P2SM special case that limits all reference matching strings within the same CU being coded and allows only SRB strings (left-matching) and reference matching strings above the current strings (above-matching). The two special cases are called big string case and SRB string only case [31] in P2SM. Since P2SM is developed in a late stage of HEVC SCC project, it is not in the HEVC SCC draft. P2SM is adopted into the AVS screen mixed content coding working draft as universal string prediction (USP) tool.



▲Figure 5. A letter S coded by P2SM technique.

5 Coding Performance Comparison of IBC, PLT, and P2SM

Coding performance comparison experiments use HM-16.6+SCM-5.2 reference software [35] and HM-16.6+P2SM software [31]. The following coding options are compared:

1) NoSCC implemented by disabling both IBC and PLT in HM-16.6+SCM-5.2
2) IBC implemented by disabling only PLT in HM-16.6+SCM-5.2
3) PLT implemented by disabling only IBC in HM-16.6+SCM-5.2
4) IBC+PLT (SCM which includes both IBC and PLT) implemented by HM-16.6+SCM-5.2
5) P2SM implemented in HM-16.6+P2SM.

The experimental results are generated under the common test conditions and lossy all-intra configuration defined in [34]. Fourteen test sequences are used in the experiment. The test sequences are classified into four categories: text and graphics with motion (TGM), mixed content (MC), camera captured (CC), and animation (ANI). YCbCr (YUV) color format version is used in the experiment. To evaluate the overall coding performance, the Bjøntegaard delta rate (BD-rate) metric [36], [37] is used. For each category, an average BD-rate reduction is calculated. Encoding and decoding software runtime are also compared for evaluating the complexity of the encoder and decoder.

Tables 1–4 show the coding performance improvement (BD-rate reduction percentage in negative numbers) of IBC, PLT, IBC+PLT (SCM), and P2SM, respectively. **Table 5** shows the coding performance improvement of P2SM over SCM.

▼Table 1. Coding performance improvement of IBC over NoSCC

	Y	U	V
TGM	-47.80%	-48.51%	-48.67%
MC	-41.23%	-42.21%	-42.34%
ANI	-1.38%	-1.72%	-1.58%
CC	-0.07%	-0.12%	-0.10%
Enc Time		158.08%	
Dec Time		78.75%	

ANI: animation Dec: Decoding MC: mixed content
CC: camera captured Enc: Encoding TGM: text and graphics with motion

▼Table 2. Coding performance improvement of PLT over NoSCC

	Y	U	V
TGM	-40.83%	-46.18%	-48.15%
MC	-24.70%	-34.70%	-34.21%
ANI	0.19%	-3.04%	-2.78%
CC	0.03%	0.02%	-0.02%
Enc Time		125.48%	
Dec Time		90.89%	

ANI: animation Dec: Decoding MC: mixed content
CC: camera captured Enc: Encoding TGM: text and graphics with motion

▼Table 3. Coding performance improvement of IBC+PLT (SCM) over NoSCC

	Y	U	V
TGM	-57.06%	-60.76%	-62.35%
MC	-45.57%	-50.85%	-50.90%
ANI	-1.17%	-4.40%	-4.08%
CC	-0.03%	-0.13%	-0.06%
Enc Time		172.19%	
Dec Time		80.85%	

ANI: animation Dec: Decoding MC: mixed content
CC: camera captured Enc: Encoding TGM: text and graphics with motion

▼Table 4. Coding performance improvement of P2SM overNoSCC

	Y	U	V
TGM	-58.19%	-61.34%	-63.00%
MC	-46.26%	-51.24%	-51.50%
ANI	-0.01%	5.39%	2.62%
CC	0.15%	1.21%	1.60%
Enc Time		140.63%	
Dec Time		80.02%	

ANI: animation Dec: Decoding MC: mixed content
CC: camera captured Enc: Encoding TGM: text and graphics with motion

▼Table 5. Coding performance improvement of P2SM over SCM

	Y	U	V
TGM	-4.74%	-4.24%	-4.20%
MC	-1.40%	-1.02%	-1.45%
ANI	1.19%	10.43%	7.00%
CC	0.17%	1.34%	1.66%
Enc Time		81.67%	
Dec Time		98.98%	

ANI: animation Dec: Decoding MC: mixed content
CC: camera captured Enc: Encoding TGM: text and graphics with motion

The experimental results show:

1) For screen content (TGM and MC categories), P2SM has higher coding performance than IBC or PLT or both combined.

2) IBC has higher coding performance than PLT, and both have significant overlap.

3) For typical and common screen content (TGM), P2SM is superior to IBC and PLT combined (HM-16.6+SCM-5.2) by close to 5% in term of BD-rate.

Recently, it is reported [32], [33] that P2SM can achieve significant coding performance improvement for screen content rendered using sub-pixel-rendering techniques such as ClearType developed and widely applied in text rendering to achieve clear and smooth text display on an LCD panel. For a ClearType snapshot and a ClearType test sequence, P2SM can achieve 39.0% and 35.4% Y BD-rate reduction, respectively, comparing to HM-16.6+SCM-5.2.

6 Conclusions

Driven by increasing demand from both existing application areas such as Wi-Fi display and emerging application areas such as cloud computing platforms, SCC technology has made significant progress in the past three years.

Two major SCC standardization projects so far are HEVC SCC project and AVS/IEEE SCC project. Both are expected to complete by the second half of 2016. Two special cases of P2SM, i.e. IBC and PLT are adopted into the HEVC SCC draft. P2SM is adopted into the AVS screen mixed content coding working draft using the name of universal string prediction (USP).

Another technique named adaptive color transform (ACT) is also adopted into HEVC SCC. ACT is based on a prediction residual coding technique [38] and is a general technique instead of SCC dedicated. ACT is mainly effective on RGB color format sequences and has negligible effect on YUV color format sequences.

String matching is a superset of block matching which has been thoroughly studied for more than thirty years. P2SM provides a flexible trade-off between coding efficiency and coding complexity. String matching technology is still in its early stage of development, much like MPEG-1 stage of block matching technology, and has significant room for improvement. Therefore, future work in SCC and general video coding includes: 1) Further study of pattern matchability in screen content pictures and other types of contents, 2) improvement on string matching technology to code a variety of contents with different pattern matchability efficiently, 3) further reduction of coding complexity of string matching techniques, and 4) Optimization of string matching techniques for specific application areas with special requirement.

References

[1] R. Joshi, S. Liu, G. Sullivan, et al., "High efficiency video coding (HEVC) screen content coding: draft 4," JCT-VC, Warsaw, Poland, JCTVC-U1005, Jun. 2015.

[2] T. Lin, K. Zhou, and S. Wang. "Cloudlet-screen computing: a client-server architecture with top graphics performance," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 13, no. 2, pp. 96–108, June 2013. doi: 10.1504/IJAHUC.2013.054174.

[3] Y. Lu, S. Li, and H. Shen, "Virtualized Screen: A Third Element for Cloud_Mobile Convergence," *IEEE Multimedia*, vol. 18, no. 2, pp. 4–11, Apr. 2011. doi: 10.1109/MMUL.2011.33.
[4] T. Lin and S. Wang. "Cloudlet-screen computing: a multi-core-based, cloud-computing-oriented, traditional-computing-compatible parallel computing paradigm for the masses," in *IEEE International Conference on Multimedia and Expo*, New York, USA, Jul. 2009, pp. 1805–1808. doi: 10.1109/ICME.2009.5202873.
[5] M. Budagavi and D. Kwon, "AHG8: video coding using Intra motion compensation," JCT-VC, Incheon, Korea, JCTVC-M0350, Apr. 2013.
[6] D. Kwon and M. Budagavi, "Intra motion compensation with variable length intra MV coding," JCT-VC, Vienna, Austria, JCTVC- N0206, Jul. 2013.
[7] C. Pang, J. Sole, L. Guo, M. Karczewicz, and R. Joshi, "Intra motion compensation with 2-D MVs," JCT-VC, Vienna, Austria, JCTVC- N0256, Jul. 2013.
[8] C. Pang, J. Sole, L. Guo, R. Joshi, and M. Karczewicz, "Displacement vector signaling for intra block copying," JCT-VC, Geneva, Switzerland, JCTVC- O0154, Oct. 2013.
[9] C. Lan, X. Peng, J. Xu, and F. Wu, "Intra and inter coding tools for screen contents," JCT-VC, Geneva, Switzerland, JCTVC-E145, Mar. 2011.
[10] W. Zhu, W. Ding, et al., "Screen content coding based on HEVC framework," *IEEE Transaction on Multimedia*, vol.16, no.5, pp. 1316–1326, Aug. 2014.
[11] L. Guo, W. Pu, et al., "Color palette for screen content coding," in *IEEE International Conference on Image Process*, Paris, France, Oct. 2013, pp. 5556–5560.
[12] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Transactions on Information Theory*, vol. 23, no. 3, pp. 337–343, May 1977.
[13] T. Lin, K. Zhou, X. Chen, and S. Wang, "Arbitrary shape matching for screen content coding," in *IEEE Picture Coding Symposium*, San Jose, USA, Dec. 2013, pp. 369–372. doi: 10.1109/PCS.2013.6737760.
[14] W. Zhu, J. Xu, W. Ding, Y. Shi, and B. Yin, "Adaptive LZMA-based coding for screen content," in *IEEE Picture Coding Symposium*, San Jose, USA, Dec. 2013, pp. 373–376. doi: 10.1109/PCS.2013.6737761.
[15] T. Lin, X. Chen, and S. Wang, "Pseudo-2-D-matching based dual-coder architecture for screen contents coding," in *IEEE International Conference on Multimedia and Expo*, San Jose, USA, Jul. 2013, pp. 1– 4. doi: 10.1109/ICMEW.2013.6618315.
[16] S. Wang and T. Lin, "Compound image compression based on unified LZ and hybrid coding," *IET Image Processing*, vol. 7, no. 5, pp. 484–499, May 2013. doi: 10.1049/iet ipr.2012.0439.
[17] T. Lin, P. Zhang, S. Wang, K. Zhou, and X. Chen, "Mixed chroma sampling-rate high efficiency video coding for full-chroma screen content," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 1, pp. 173–185, Jan. 2013. doi: 10.1109/TCSVT.2012.2223871.
[18] W. Zhu, W. Ding, R. Xiong, Y. Shi, and B. Yin, "Compound image compression by multi-stage prediction," in *IEEE Visual Communications and Image Processing Conference*, San Diego, USA, Nov. 2012, pp. 1–6. doi: 10.1109/VCIP.2012.6410758.
[19] S. Wang and T. Lin, "United coding method for compound image compression," *Multimedia Tools and Applications*, vol. 71, no. 3, pp. 1263–1282, 2014. doi: 10.1007/s11042-012-1274-y.
[20] S. Wang and T. Lin, "United coding for compound image compression," in *IEEE International Conference on Image and Signal Processing*, Yantai, China, Oct. 2010, pp. 566–570. doi: 10.1109/CISP.2010.5647270.
[21] S. Wang and T. Lin, "A unified LZ and hybrid coding for compound image partial-lossless compression," in *IEEE International Conference on Image and Signal Processing*, Cairo, Egypt, Oct. 2009, pp. 1– 5. doi: 10.1109/CISP.2009.5301019.
[22] W. Ding, Y. Lu, and F. Wu, "Enable efficient compound image compression in h.264/AVC intra coding," in *IEEE International Conference on Image Processing*, Penang, Malaysia, vol. 2, Oct. 2007, pp. 337– 340. doi: 10.1109/ICIP.2007.4379161.
[23] L. Zhao, K. Zhou, and T. Lin, "CE3: results of test B.4.2 (minimum string length of 20 pixels) on intra string copy," JCT-VC, Geneva, Switzerland, JCTVC-T0136, Feb. 2015.
[24] C. Hung, Y. Chang, J. Tu, C. Lin, and C. Lin, "CE3: crosscheck of CE3 test B.4.2 (JCTVC-T0136)," JCT-VC, Geneva, Switzerland, JCTVC-T0179, Feb. 2015.
[25] L. Zhao, K. Zhou, S. Wang, and T. Lin, "Non-CE3: improvement on intra string copy," JCT-VC Doc JCTVC-T0139, Feb. 2015.
[26] R. Liao, C. Chen, W. Peng, and H. Hang, "Crosscheck of non-CE3: improvement on intra string copy (JCTVC-T0139)," JCT-VC, Geneva, Switzerland, JCTVC-T0200, Feb. 2015.
[27] T. Lin, K. Zhou, and L. Zhao, "Non-CE1: enhancement to palette coding by palette with pixel copy (PPC) coding," JCT-VC, Warsaw, Poland, JCTVC-U0116, Jun. 2015.
[28] L. Zhao, W. Cai, J. Guo, and T. Lin, "Flexible coding tools to significantly improve SCC performance in cloud and mobile computing," JCT-VC, Warsaw, Poland, JCTVC-U0189, Jun. 2015.
[29] R. Liao, C. Chen, W. Peng, et al., "Crosscheck of Non-CE1: Enhancement to palette coding by palette with pixel copy (PPC) coding," JCT-VC, Warsaw, Poland, JCTVC-U0173, Jun. 2015.
[30] W. Wei, X. Meng, "Cross-check report of U0116," JCT-VC, Warsaw, Poland, JCTVC-U0189, June 2015.
[31] K. Zhou, L. Zhao, and T. Lin, "Advanced SCC tool using Pseudo 2D string matching (P2SM) integrated into HM16.6," JCT-VC, Geneva, Switzerland, JCTVC-V0094, Oct. 2015.
[32] L. Zhao, J. Guo, and T. Lin, "Significantly improving coding performance of Clear Type texts and translucently blended screen content by P2SM," JCT-VC, Geneva, Switzerland, JCTVC-V0095, Oct. 2015.
[33] J. Guo, L. Zhao, and T. Lin, "A new SCC test sequence with ClearType text rendering for consideration," JCT-VC, Geneva, Switzerland, JCTVC-V0097, Oct. 2015.
[34] H. Yu, R. Cohen, K. Rapaka, and J. Xu, "Common conditions for screen content coding tests," JCT-VC, Warsaw, Poland, JCTVC-U1015, Jun. 2015.
[35] Heinrich Hertz Institute. (2015). *Rec. ITU-T H.265|ISO/IEC 23008-2 High Efficiency Video Coding* [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.6+SCM-5.2
[36] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," ITU-T SG16 Q.6 Document, VCEG-M33, Austin, US, Apr. 2001.
[37] G. Bjøntegaard, "Improvements of the BD-PSNR model," ITU-T SG16 Q.6 Document, VCEG-AI11, Berlin, Germany, Jul. 2008.
[38] D. Marpe, H. Kirchhoffer, V. George, P. Kauff, and T. Wiegand, "Macroblock-adaptive residual color space transforms for 4:4:4 video coding," in *IEEE International Conference on Image Processing*, Atlanta, USA, 2006, pp. 3157–3160.

Manuscript received: 2015-11-12

Biographies

LIN Tao (lintao@tongji.edu.cn) received his MS and PhD degrees from Tohoku University, Japan in 1985 and 1989. He has been with VLSI Lab, Tongji University, China since 2003. In 2005, he was awarded "Chang Jiang Scholars", the highest honor given by China Ministry of Education. From 1988 to 2002, he was a postdoctoral researcher with University of California, Berkeley, and developed multimedia ICs and products at several companies in Silicon Valley, including Integrated Device Technology, Inc., PMC-Sierra Inc., Cypress Semiconductor Corp., and NeoMagic Corp. He has been granted 24 US patents and 14 China patents. His current research interests include cloud-mobile computing, digital signal processing, audiovisual coding, and multimedia SoC design.

ZHAO Liping (vlsi@tongji.edu.cn) received her MS degree in computer science and technology from Hunan University, China in 2009. She is currently a PhD candidate in control science and engineering at VLSI lab of Tongji University, China. Her current research interests include screen content coding and video coding.

ZHOU Kailun (vlsi@tongji.edu.cn) received his MS degree from Shanghai Jiaotong University, China in 2003. He is currently pursuing the PhD degree with Tongji University, China. His current research interests include embedded system design, video coding, and ASIC architecture, design and verification.

DOI: 10.3969/j. issn. 1673-5188. 2016. S0. 008
http://www.cnki.net/kcms/detail/34.1294.TN.20160603.1100.002.html, published online June 3, 2016

Human Motion Recognition Based on Incremental Learning and Smartphone Sensors

LIU Chengxuan[1], DONG Zhenjiang[2], XIE Siyuan[2], and PEI Ling[1]
(1. Shanghai Jiao Tong University, Shanghai 200240, China;
2. ZTE Corporation, Shenzhen 518057, China)

Abstract

Batch processing mode is widely used in the training process of human motion recognition. After training, the motion classifier usually remains invariable. However, if the classifier is to be expanded, all historical data must be gathered for retraining. This consumes a huge amount of storage space, and the new training process will be more complicated. In this paper, we use an incremental learning method to model the motion classifier. A weighted decision tree is proposed to help illustrate the process, and the probability sampling method is also used. The results show that with continuous learning, the motion classifier is more precise. The average classification precision for the weighted decision tree was 88.43% in a typical test. Incremental learning consumes much less time than the batch processing mode when the input training data comes continuously.

Keywords

human motion recognition; incremental learning; mapping function; weighted decision tree; probability sampling

1 Introduction

Human motion recognition involves recognizing what a person is doing and is an important aspect of context awareness [1]. It can be used for diverse purposes, such as ubiquitous computing, sports training, virtual reality, and health care. A promising application is remote monitoring of elderly people who live alone and need support. An emergency situation arising from a fall could be detected and responded to quickly [2]. Recently, sports bracelets that detect a person's motion have become very popular. Such bracelets can calculate how many calories a person consumes in a day and give reminders about healthy lifestyle. Human motion recognition has also been introduced into personal navigation to help increase the location accuracy [3]–[5].

Methods of motion recognition and classification include computer vision and inertial sensor data processing. Early research on motion recognition has focused on vision-based systems with one or more cameras [6], [7]. A camera system is practical when motion is confined to a limited area, such as an office or house, and the environment is well-lit. However, when a person is moving from place to place, a camera system is much less convenient because it cannot move in the same way a person does. In terms of privacy, a vision-based system puts a degree of psychological pressure on a person and causes them to act unnaturally. As well as vision-based solutions, sensor-based solutions are also extensively used to study human motion [8]–[11]. Most previous research on motion recognition has assumed that inertial sensors are fixed on the human body in a known orientation [12], [13]. In a well-cited work [14], multiple accelerometer sensors worn on different parts of the human body detect common activities, such as sitting, standing, walking or running. In [15], a small low-power sensor board is mounted at a single location on the body. Then, a hybrid approach is taken to recognize activities. This approach combines the boosting algorithm, which discriminatively selects useful features, and HMMs, which capture the temporal regularities and smoothness of activities. However, the assumption made in laboratory experiments usually cannot be made in a regular mobile environment. In some other research, a phone has been used as the sensor to collect motion data for offline analysis [16], [17]. In [18], a phone-centric sensing system is described. The position of the mobile phone on the human body is assumed to be fixed—e.g., in a pocket, clipped to a belt, or on a lanyard—and an inference model is trained according to the phone position. Compared to image-processing-based motion recognition, inertial-sensor-based motion recognition is cheaper, less limited by the environment, and involves smaller devices. Such sensors have already been integrated into smartphones, which are developing rapidly. Therefore, inertial-sensor-based motion recognition may be more popular in the future.

To recognize human motion, a classifier should be modeled. In the training process, an algorithm can be divided into batch-processing mode and incremental-learning mode. In batch-processing mode, all the history training data is used to model the motion classifier. When the new training data is available and the classifier needs to be updated, all the history data must be gathered again to retrain the classifier. Therefore, all the train-

This work is partly supported by the National Natural Science Foundation of China under Grant 61573242, the Projects from Science and Technology Commission of Shanghai Municipality under Grant No. 13511501302, No. 14511100300, and No. 15511105100, Shanghai Pujiang Program under Grant No. 14PJ1405000, and ZTE Industry-Academia-Research Cooperation Funds.

ing data must be preserved. This means that a huge amount of storage space is needed. As the amount of input training data increases, the training process becomes more complex and takes longer. Most current research on vision- and inertial-sensor-based human motion recognition focuses on this processing mode. Incremental learning only uses new incoming training data to update the classifier model; therefore, updating is more efficient, history data does not need to be stored, and much free space is spared. In [19], the authors propose pattern recognition based on neural networks and learning new chunks of patterns while keeping the previous ones intact. In [20], the authors suggest Learn ++ , an approach inspired by Adaboost. This approach is based on neural - network - based ensemble classifiers working on a digital optics database. In [21], a Gaussian mixture model and resource allocation for learning were applied in the context of a dormitory to study the habits of students. In [22], the authors propose an approach to incrementally learning the relationship between motion primitives in order to form longer behaviors. In general, when the training data set is huge and data is continuously coming in, incremental learning mode is a better choice.

The main goal of our research is to provide a complete solution for human motion recognition based on incremental learning and smartphone inertial sensors. We illustrate the flow of motion pattern recognition and typical motion feature sets. We also show how to apply incremental learning to human motion recognition in detail. At the same time, we develop a weighted decision tree for the incremental learning framework.

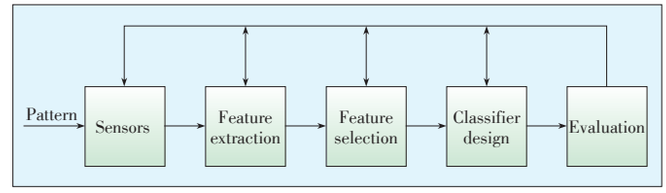
The remainder of this paper is organized as follows. In section 2, we describe the general pattern-recognition framework, date preprocessing, and feature extraction. In section 3, we discuss the incremental learning method used for human motion recognition. In section 4, we discuss the experimental results. In section 5, we make concluding remarks.

2 Motion Recognition

2.1 Pattern-Recognition Framework

Pattern recognition is a subject dealing with object classification and recognition. It encompasses face identification, character recognition, and voice recognition. It also encompasses human motion recognition. A general pattern - recognition system framework is shown in **Fig. 1**. The components of this framework are defined in **Table 1**.

Because raw sensor data is just a series of waves, it cannot be directly used for pattern recognition. Therefore, pattern features are extracted in order to describe the patterns. A pattern may have many features; however, more features do not mean better performance. If different patterns cannot discriminate when the chosen features are applied, the designed classifier performs badly. Feature selection (feature compression) is an important part of pattern recognition. With a well-selected feature subset, the generated classifier has a higher classification rate. Calculation complexity is also greatly reduced. The classifier is designed after the feature-selection process is completed. There are two methods for training a classifier: supervised learning (also called supervised clustering) and unsupervised learning. The main difference between these methods is that the labels of instances in supervised learning are known to the learner. This is not the case for unsupervised learning. The performance of the classifier is evaluated using test instances. The components shown in Table 1 are not independent. To improve overall performance, one component may feed back to the previous component, and the previous component is designed again.



▲Figure 1. Pattern-recognition system framework.

▼Table 1. Component definition of pattern recognition system

Component	Definition
Sensors	Obtain raw sensor data
Feature extraction	Extraction of features from raw sensor data
Feature selection	Selection of best feature subset
Classifier design	Modeling of classifier using training data
Evaluation	Evaluation of the performance of classifier model

In this paper, we use the general pattern-recognition framework to recognize human motion. We mainly focus on the classifier design component; the feature extraction and feature selection components are merged into one procedure.

2.2 Motion Definitions

Common human motions include keeping still, walking, running, climbing stairs, using an elevator, driving, and taking a bus. The possible motion set differs in different scenarios. In this paper, we limit the scenario to an office. The motions related to this scenario are shown in **Table 2**.

2.3 Data Collection and Preprocessing

In this paper, a three-axis linear smartphone accelerometer

▼Table 2. Motion state definition

Motion state	Definition
M1	Still
M2	Walking
M3	Climbing up stairs
M4	Climbing down stairs
M5	Running

is used to collect the raw motion data of a person. In fact, instead of the total acceleration, the linear acceleration infers the motions of a human. The collection process is controlled by an application we developed. Through a simple graphic user interface, we can start or stop collecting. The sensor sampling rate is 50 Hz, so we obtain 50 raw samples per second.

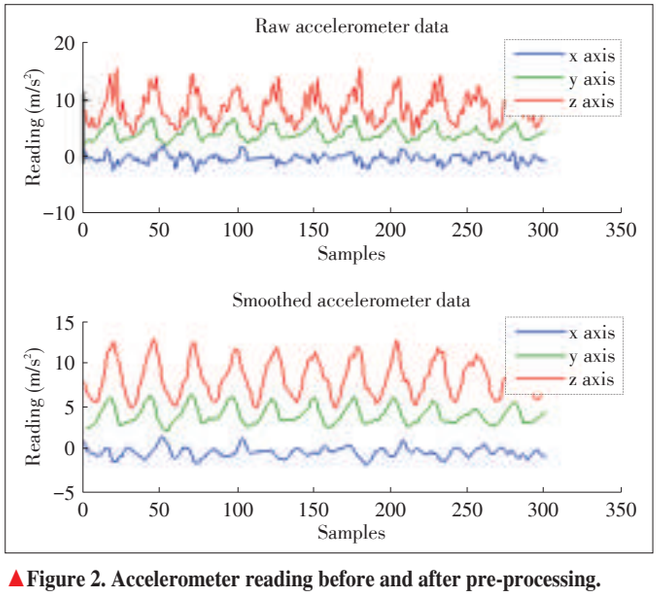
Because there are some noises in the raw sensor data, a five-stage moving window is used to eliminate them. **Fig. 2** shows the output of the moving window is smoother than the raw linear accelerometer data.

2.4 Feature Extraction

Features from the linear acceleration are used for motion recognition. In most of the previous work involving inertial sensors for motion recognition, the sensors are mounted on the human body in a known orientation and position. However, this is not practical in real life because of the flexible use of smartphones. To avoid the orientation problem, instead of directly using the features from x, y, z axes, the vertical and horizontal components of the linear acceleration are extracted by projecting the linear acceleration vector to the gravity vector. Some smartphones have the gravity sensor imbedded directly. With others, the gravity can be obtained using the following method. First, the smartphone is kept still for a few seconds. Then, the averages of the three axis readings are calculated as the approximation of the gravity vector [23].

Suppose $\vec{a}=(a_x,a_y,a_z)$ is the linear acceleration vector and $\vec{g}=(g_x,g_y,g_z)$ is the gravity vector. Then the vertical component of linear acceleration is $\vec{a}_v=\left(\frac{\vec{a}\times\vec{g}}{\vec{g}\times\vec{g}}\right)\vec{g}$ and the horizontal component of the linear acceleration is $\vec{a}_h=\vec{a}-\vec{a}_v$.

Both time and frequency domain features are extracted to construct the feature vector. In the time domain, mean, variance, skewness, kurtosis, and so on are the features used. The formulas for these features are shown in **Table 3**. In the frequency domain, an FFT algorithm is applied, and first dominant frequency, second dominant frequency, and the amplitude of the first and second dominant frequencies are the features used. All the features used are shown in **Table 4**. The instance window for extracting features is 2 s, and a 50% overlapping window is used to obtain feature vectors more efficiently. This has been demonstrated to be useful [24].



▲Figure 2. Accelerometer reading before and after pre-processing.

3 Incremental Learning

Recently, incremental learning has been popular in data mining and has attracted the attention of both academia and industry. Many of today's data-intensive computing applications require an algorithm that is capable of incrementally learning from large-scale dynamic data. An incremental learning algorithm learns new knowledge continuously over time and up-

▼Table 3. Statistical feature definitions

Feature	Definition
Mean	$\bar{m}=\frac{1}{N}\sum_{i=1}^{N}x_i$
Variance	$\sigma^2=\frac{1}{N}\sum_{i=1}^{N}(x_i-\bar{m})^2$
Skewness	$skew=\frac{1}{N\sigma^3}\sum_{i=1}^{N}(x_i-\bar{m})^3$
Kurtosis	$kur=\frac{1}{N\sigma^4}\sum_{i=1}^{N}(x_i-\bar{m})^4$

▼Table 4. Human motion feature definitions

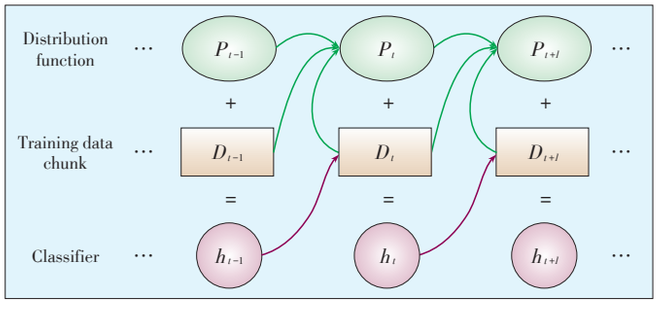
Feature	Definition
linaccM_mean	Mean of module of linear acceleration
linaccM_var	Variance of module of linear acceleration
linaccM_skew	Skewness of module of linear acceleration
linaccM_kur	Kurtosis of module of linear acceleration
linaccV_mean	Mean of vertical component of linear acceleration
linaccV_var	Variance of vertical component of linear acceleration
linaccV_skew	Skewness of vertical component of linear acceleration
linaccV_kur	Kurtosis of vertical component of linear acceleration
linaccH_mean	Mean of horizontal component of linear acceleration
linaccH_var	Variance of horizontal component of linear acceleration
linaccH_skew	Skewness of horizontal component of linear acceleration
linaccH_kur	Kurtosis of horizontal component of linear acceleration
firstfreq	The first dominant frequency
firstpeak	The amplitude of the first dominant frequency
secondfreq	The second dominant frequency
secondpeak	The amplitude of the second dominant frequency
energy	The mean energy of linear acceleration

dates the knowledge system to benefit future learning and decision-making.

In the human motion recognition domain, most researchers have used batch processing methods to recognize human motion. A classifier that uses batch processing keeps an invariable knowledge system after the learning. Usually, the classifier cannot learn new knowledge directly in order to expand itself unless all historical training data is gathered together with the new training data. Compared with incremental learning, batch mode learning, which updates itself, requires much storage space for historical training data, and the new training process is slower.

3.1 Framework for Incremental Learning

Inspired by the classification performance in [25], we use a similar incremental learning framework for human motion recognition. **Fig. 3** gives an overview of the algorithm. The learner is continuously presented with the training data flows over



▲**Figure 3. Incremental learning overview.**

time. The previous knowledge in this case includes the hypothesis, which was developed at time $t-1$, and the distribution function P_{t-1} is applied to the data set D_{t-1}. For the first block of the received training data, the initial distribution function P_1 is given a uniform distribution because nothing has been learned yet. As data blocks continually come in, a series of distribution functions is developed to represent the learning capability of the data in each block. Based on distribution function P_t, a hypothesis h_t, which is a human motion classifier, can be developed. In this classifier, the decision boundary is automatically forced to focus more on difficult-to-learn regions. When P_t and h_t have been obtained, the system uses its knowledge to facilitate learning from the next block of training data D_{t+1}.

Algorithm 1 can be divided into two parts. First, a mapping function estimates the initial distribution function of the current training data using the last input training data set and its corresponding distribution function. Second, a classifier using training data with probability distribution function is developed.

Algorithm 1. Incremental learning algorithm

Input: Sequence of data chunks $D_t, (t=1,2,3,...)$, each data chunk includes some instances. An instance is composed of a feature vector and a label.

Learning process:

1. For the first data set, suppose there are m samples, $D_1=\{x_i,y_i\}, (i=1,2,...,m_1)$, where x_i is the feature vector of the i-th sample and y_i is the label of the i*th* sample. The initial distribution function P_1 is a uniform distribution.

2. Train a classifier h_1 using the training data set D_1 and its corresponding probability distribution function P_1

3. For $t \geqslant 2$, use the last data set D_{t-1} and distribution function P_{t-1} to estimate the distribution function $\hat{P}_t$ of the new training data set D_t

4. Apply the last classifier h_{t-1} to the new data set D_t, calculate the pseudo error e_t:

$$e_t = \sum_{\substack{j:h_{t-1}(x_j)\neq y_j \\ (x_j,y_j)\in D_t}} \hat{P}_t(j) .$$

5. Set $\beta_t = e_t/((n-1)(1-e_t))$, n is the number of all different labels.

6. Update the distribution function of D_t, the misclassified sample will get a higher probability or weight:

$$P_t(j) = \frac{\hat{P}_t(j)}{Z_t}\begin{cases}\beta_t & if\ h_{t-1}(x_j)=y_j \\ 1 & otherwise\end{cases},$$

where Z_t is a normalization constant so that P_t is a distribution and $\sum_{j=1}^{m_t} P_t(j)=1$.

7. Model the classifier h_t using the training set D_{t+1} and its corresponding distribution function P_t

8. When new training data set comes, go back to 3 and repeat the procedure.

Output: After T classifiers have been trained, we obtain T basic classifiers and their corresponding weights. The final output is obtained by the combination of these. Here, $\log(1/\beta_t)$ is the weight of each basic classifier:

$$h_{final}(x) = \arg\max_{y\in Y} \sum_{t:h_T(x)=y} \log(1/\beta_t) .$$

3.2 Mapping Function Design

The mapping function is an important component in the incremental learning framework. It connects past experience to the newly received data and adapts such knowledge to the data sets received in future. Nonlinear regression models can be used as well as mapping functions. Here, we consider support vector regression (SVR) [26]. Suppose $y=f(x)$ is the estimated initial weight for instance x:

$$f(x)=<s,x>+b ,$$

where s and b are the slope and intercept of the linear estima-

tion function $f(x)$, respectively. To obtain an accurate estimation $f(x)$, a convex optimization problem can be extracted:

$$(s_t,b_t)=\arg\min_{s_t\in R^n,b_t\in R}\left(\sum_{i=1}^{m_{t-1}}\|y_i-<s_t,x_i>-b_t\|^2\right).$$

Alternative strategies can be used as well. For example, other types of regression model, such as multilayer perceptron or regression tree, can be integrated into the incremental learning framework according the application requirements.

3.3 Hypothesis Based on Probability-Distributed Training Data

In the conventional classifier design process, all instances in the training data set have the same weight. This means that instances that are difficult to classify are treated the same as those that are not. However, instances that are difficult to classify should be weighted more heavily so that they will be more easily recognized in the newly designed classifier.

Here, we introduce two methods for classifier modeling, both of which use the probability distributed training data. With the first method, the probability is used as the weight of the instance in the calculation of the decision boundary. We illustrate this method in the proposed weighted decision-tree algorithm. With the second method, the probability distribution function is used as the sampling probability. All training instances are given a sampling probability and sampled into the real training set. The higher the instance's probability, the easier it is added to the real training set.

3.3.1 Weighted Decision Tree

Decision tree is a classic algorithm used in pattern recognition and machine learning. It uses the information entropy gain to split training data, and it constructs a tree to represent the classifier. The information entropy of a tree node in the decision-tree algorithm is given by:

$$I(t)=-\sum_{i=1}^{M}P(\omega_i|t)\log_2 P(\omega_i|t),$$

where t is the current node, and $P(\omega_i|t)$ is the probability of the class ω_i in node t. In a conventional decision tree, it is N_{ω_i}/N_t, in which N_{ω_i} is the number of instances belonging to class ω_i, and N_t is the total number of instances in node t.

When node t is split, maximum entropy gain criterion is used. The formula is:

$$\Delta I(t)=I(t)-\frac{N_{t_l}}{N_t}I(t_l)-\frac{N_{t_r}}{N_t}I(t_r),$$

where t_l is the left child node, N_{t_l} is the number of the instances in the left child node, t_r is the right child node, and N_{t_r} is the number of instances in the right child node. In the conventional decision tree, all the instance have the same weight, i.e., $1/N_t$.

In this paper, we propose a weighted decision tree in which each training instance is given a different weight instead of $1/N_t$ (the original decision tree is a special case of the weighted decision tree). Thus, the heavier the weight of an instance, the bigger the information entropy of its class. This forces the decision boundary to focus on the more heavily weighted instances, which are difficult to learn. Thus, in the next incoming data block, the instance that is difficult to learn is weighted more heavily, and the final classifier will recognize it more easily.

In the information entropy formula, $P(\omega_i|t)$ is the sum of the weights of the instances in class ω_i. In the split formula, the factor $I(t_l)$ is the sum of the weights of instances in the left child node. The factor $I(t_r)$ is the sum of the weights of instances in the right child node. Thus, the instance that is difficult to learn will be weighted more heavily, and the instance that is easy to learn weighted more lightly.

3.3.2 Sampling Probability Function

With the sampling probability function method, all training data is sampled into the real training set according to their probability. The instance which is hard to learn will have a higher probability to be sampled into the real training set. However, low probability instance will have a lower probability to be chosen into the final training set. That means there may exist several duplicates of the hard instance in the real training set, and easy instance may not exist in the final training set. More basic classifier category can be contained in this framework.

4 Experiments and Evaluation

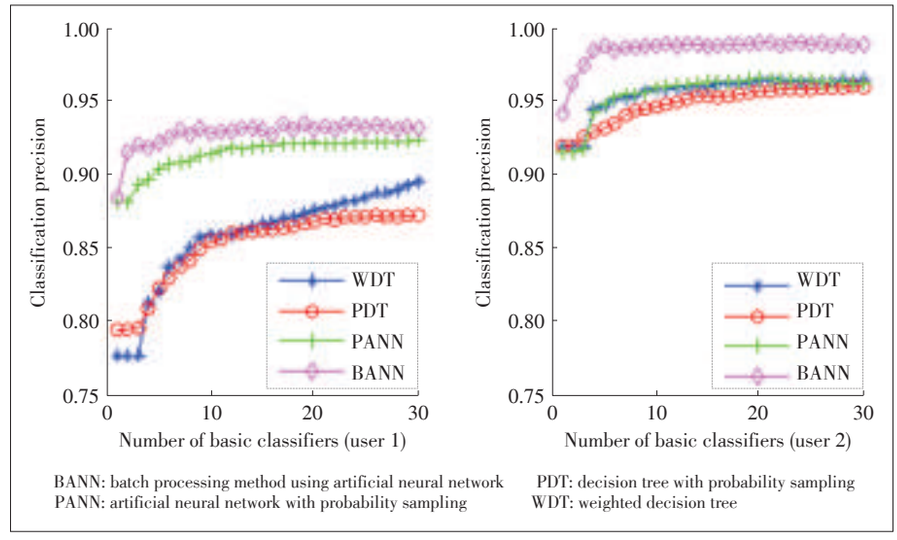
The device used in our experiments is a Google Nexus 5. This smartphone has a built-in tri-axial MPU 6515 accelerometer that records the user's raw motion data. The Android system uses a filtering algorithm to extract the linear accelerometer and gravity accelerometer. Experimental data was collected from two males. One was 1.85 m tall and weighed 70 kg. The other was 1.75 m tall and weighed 65 kg. Both males stood still, walked, ran, and climbed up and down stairs in and around our office building. The smartphone was held in the hand with the phone screen facing upwards. All sensor data was stored and processed offline.

4.1 Classification Precision Test

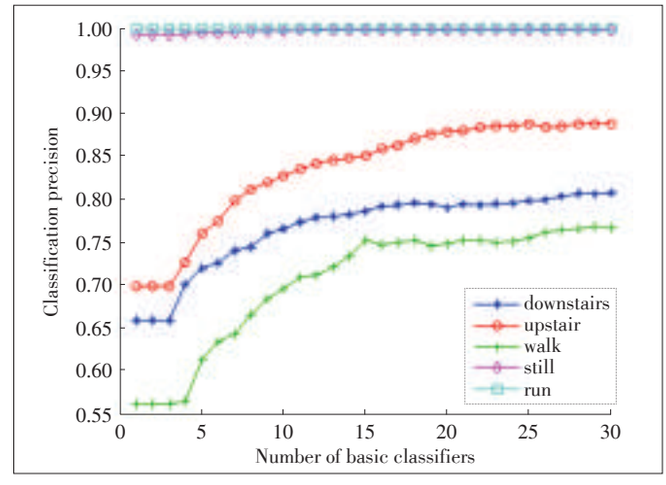
To determine the performance of incremental learning, the data was split into many blocks. In our test, there were 250 instances in a block; however, it is not essential to have the same number of instances in each block. The data blocks were input into the incremental-learning framework to simulate the continuous learning process. The two important parts of the incremental learning process are mapping function design and motion classifier modeling based on probability-distributed training

data. An artificial neural network was used for the mapping function because such a network is well integrated into the MATLAB toolbox. Weighted decision tree and sampling probability methods were used for classifier modeling. At the same time, batch processing method using an artificial neural network is used for the comparison.

The results of classification based on different learning methods are shown in **Fig. 4**. Incremental learning using three kinds of hypothesis strategy—weighted decision tree (WDT), decision tree with probability sampling (PDT), and artificial neural network with probability sampling (PANN) —results in a higher classification rate with continuous learning. For both user 1 and user 2 probability sampling, PANN performs better than PDT. When WDT is used for user 1, classification precision is only slightly higher than that of PDT when the number of basic classifiers is large enough. However, when WDT is applied to user 2, it performs better than both PDT and PANN, both of which use probability sampling. The increasing curves show that the incremental learning algorithm learns the new knowledge continuously and benefits future decision making. The classification rate of the batch processing method using artificial neural network (BANN) also increases when there is more training data. Both incremental learning and batch learning tend to be stable when there is enough training data. For example, PANN for user 2 needs be completed about ten times to be stable in this scenario. Because batch processing uses the training data sufficiently each time, it has a higher classification precision than incremental learning. Incremental learning using some strategy performs almost as well as batch processing. For user 1, the classification precision of PANN is slightly less than that of BANN (Fig. 4).



▲Figure 4. Classification precision of incremental learning and batch learning.



▲Figure 5. Classification precision of each motion state.

Fig. 5 shows how the classification of each motion changes for incremental learning. Take the incremental learning process of user 1 with WTD for example.

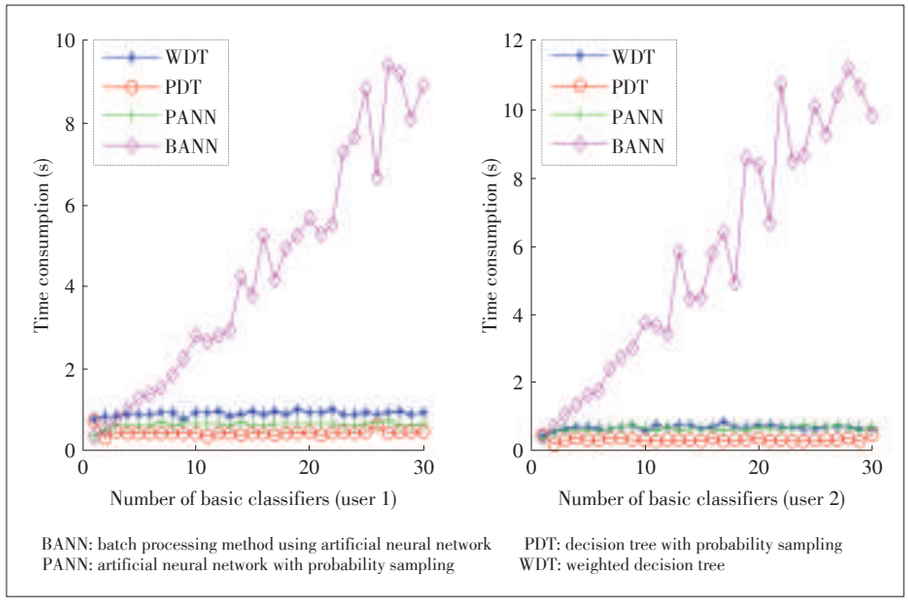
Because still and run have different feature spaces than other motions, they can be easily recognized. However, according to the smartphone inertial sensors, the person may appear to be doing similar motions when going downstairs, upstairs or walking. These three motions belong to classes of motions that are difficult to learn in this scenario. With continuous learning, the former basic classifier delivers its learned knowledge to the next classifier and forces the decision boundary to focus on the three hard-to-learn regions. Therefore, the hard-to-learn motions will have higher classification precision after learning. The confusion matrix after incremental learning is shown in **Table 5.** The average of the motion classification precision is 88.43%, which is high enough for many applications.

4.2 Computational Complexity Test

Although batch processing may be more precise than incremental learning, a huge amount of storage space and more complex computation are required. Incremental learning does not require historical training data to be stored. The computational complexity is shown in **Fig. 6**. The more complex the computa-

▼Table 5. Motion classification confusion matrix

Motion	Downstairs	Upstairs	Walking	Still	Running	Precision
Downstairs	115	13	17	2	0	80.60%
Upstairs	7	160	24	0	0	83.77%
Walking	21	19	140	0	0	77.78%
Still	0	0	0	226	0	100%
Running	0	0	0	0	202	100%



▲Figure 6. Time consumption of incremental learning and batch processing mode.

tion is, the more time the process requires. Therefore, the amount of time needed to complete the process reflects computational complexity. As training data is continuously input, the training process time is recorded for both incremental learning and batch processing modes (Fig. 6).

Regardless of which hypothesis which strategy used, the time required for incremental learning remains approximately constant, and training data is continually input. However, the amount of time needed for batch mode learning is linear to the amount of training data. With incremental learning only the new incoming data needs to be disposed. No historical training data needs to be stored or used in the new basic classifier training process, and the time to complete the process only needs to be approximately constant. Batch processing requires historical data to be stored and used to train the new classifier. Each time new training data arrives, the overall amount of training data increases linearly. Therefore, the time required in batch processing mode increases linearly at the same time. The average time consumption for both users is shown in **Table 6**. The time needed in batch processing mode BANN is several times that in incremental learning mode for both users. Because the time needed in batch processing mode increases linearly as data continually arrives, this ratio continues to increase.

▼Table 6. Average time consumption using different learning methods

	WDT	PDT	PANN	BANN
User 1	0.889 s	0.422 s	0.613 s	4.414 s
User 2	0.658 s	0.305 s	0.620 s	5.692 s

BANN: batch processing method using artificial neural network
PANN: artificial neural network with probability sampling
PDT: decision tree with probability sampling
WDT: weighted decision tree

5 Conclusion

In this paper, we have used the incremental learning method to recognize human motion. First, a mapping function was used to deliver learned knowledge to incoming data. Then, a basic classifier is modeled according to the training data with distribution. A weighted decision tree was proposed to illustrate hypothesis construction, and a sampling probability method was also employed. With continuous incoming data, the incremental learning method almost has the same classification precision as batch processing method. However, the incremental learning method has lower computational complexity and is much faster in the training phase. Considering the tradeoff between classification precision and training time, incremental learning is better than batch processing when there is huge amount of input data.

Acknowledgement

The authors would like to thank the editors and the reviewers for their very helpful comments and review. The authors are also grateful to the team members in the Shanghai Key Laboratory of Navigation and Location Based Services of Shanghai Jiao Tong University.

References

[1] L. Pei, R. Chen, J. Liu, et al., "Using motion-awareness for the 3D indoor personal navigation on a Smartphone," in *Proc. 24rd International Technical Meeting of the Satellite Division of the Institute of Navigation*, Portland, USA, Sept. 2011, pp. 2906–2912.

[2] K. Altun, B. Barshan, and O. Tunçel, "Comparative study on classifying human activities with miniature inertial and magnetic sensors," *Pattern Recognition*, vol. 43, no. 10, pp. 3605–3620, Oct. 2010. doi: 10.1016/j.patcog.2010.04.019.

[3] C. Liu, L. Pei, J. Qian, et al., "Sequence-based motion recognition assisted pedestrian dead reckoning using a smartphone," in *6th China Satellite Navigation Conference (CSNC)*, Xi'an, China, 2015, pp. 741–751,. doi: 10.1007/978-3-662-46632-2_64.

[4] L. Pei, R. Chen, J. Liu, et al., "Motion recognition assisted indoor wireless navigation on a mobile phone," in *Proc. 23rd International Technical Meeting of The Satellite Division of the Institute of Navigation*, Portland, USA, Sept. 2010, pp. 3366–3375.

[5] J. Qian, L. Pei, J. Ma, R. Ying, and P. Liu, "Vector graph assisted pedestrian dead reckoning using an unconstrained smartphone," *Sensors*, vol. 15, no. 3, pp. 5032–5057, Mar. 2015. doi: 10.3390/s150305032.

[6] T. B. Moeslund and E. Granum, "A survey of computer vision-based human motion capture," *Computer Vision and Image Understanding*, vol. 81, no. 3, pp. 231–268, Mar. 2001. doi: 10.1006/cviu.2000.0897.

[7] L. Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis," *Pattern Recognition*, vol. 36, no. 3, pp. 585–601, Mar. 2003. doi: 10.1016/S0031

-3203(02)00100-0.
[8] T. Y. Chung, Y. M. Chen, and C. H. Hsu, "Adaptive momentum-based motion detection approach and its application on handoff in wireless networks," *Sensors*, vol. 9, no. 7, pp. 5715–5739, Jul. 2009. doi: 10.3390/s90705715.
[9] D. T. P. Fong and Y. Y. Chan, "The use of wearable inertial motion sensors in human lower limb biomechanics studies: a systematic review," *Sensors*, vol. 10, no. 12, pp. 11556–11565, Dec. 2010. doi: 10.3390/s101211556.
[10] L. Pei, R. Guinness, R. Chen, et al., "Human behavior cognition using smartphone sensors," *Sensors*, vol. 12, no. 2, pp. 1402– 1424, Jan. 2013. doi: 10.3390/s130201402.
[11] R. Chen, T. Chu, K. Liu, J. Liu, and Y. Chen, "Inferring human activity in mobile devices by computing multiple contexts," *Sensors*, vol. 15, no. 9, pp. 21219–21238, Aug. 2015. doi: 10.3390/s150921219.
[12] B. Musleh, F. García, J Otamendi, et al., "Identifying and tracking pedestrians based on sensor fusion and motion stability predictions," *Sensors*, vol. 10, no. 9, pp. 8028–8053, Aug. 2010. doi: 10.3390/s100908028.
[13] W. Chen, Z. Fu, R. Chen, et al., "An integrated GPS and multi-sensor pedestrian positioning system for 3D urban navigation," *2009 Joint Urban Remote Sensing Event*, Shanghai, China, May 2009, pp. 1– 6. doi: 10.1109/URS.2009.5137690.
[14] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," *Pervasive Computing*, vol. 3001, pp. 1–17, Apr.2004,. doi: 10.1007/978-3-540-24646-6_1.
[15] J. Lester, T. Choudhury, N. Kern, et al., "A hybrid discriminative/generative approach for modeling human activities," in *IJCAI*, Edinburgh, UK, 2005, pp. 766–772.
[16] J. Yang, "Toward physical activity diary: motion recognition using simple acceleration features with mobile phones," in *Proc. 1st ACM International Workshop on Interactive Multimedia for Consumer Electronics*, Beijing, China, Oct. 2009, pp. 1–10. doi: 10.1145/1631040.1631042.
[17] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, vol. 12, no.2, pp. 74–82, Dec. 2011. doi: 10.1145/1964897.1964918.
[18] E. Miluzzo, N. D. Lane, K. Fodor, et al., "Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application," in *Proc. 6th ACM Conference on Embedded Network Sensor Systems*, Raleigh, USA, Nov. 2008, pp. 337–350. doi: 10.1145/1460412.1460445.
[19] S. Ozawa, S. Pang, and N. Kasabov, "Incremental learning of chunk data for online pattern classification systems," *IEEE Transactions on Neural Networks*, vol. 19, no. 6, pp. 1061–1074, Mar. 2008. doi: 10.1109/TNN.2007.2000059.
[20] R. Polikar, L. Updа, S. S. Updа, and V. Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 31, no. 4, pp. 497–509, Nov. 2001. doi: 10.1109/5326.983933.
[21] A. Bouchachia, M. Prossegger, and H. Duman, "Semi-supervised incremental learning," in *IEEE International Conference on Fuzzy Systems (FUZZ)*, Barcelona, Spain, Jul. 2010, pp. 1–6. doi: 10.1109/FUZZY.2010.5584328.
[22] D. Kulić, C. Ott, D. Lee, J. Ishikawa, and Y. Nakamura, "Incremental learning of full body motion primitives and their sequencing through human motion observation," *The International Journal of Robotics Research*, Nov. 2011. doi: 10.1177/0278364911426178.
[23] L. Pei, J. Liu, R. Guinness, et al., "Using LS-SVM based motion recognition for smartphone indoor wireless positioning," *Sensors*, vol. 12, no. 5, pp. 6155–6175, May 2012. doi: 10.3390/s120506155.
[24] N. Ravi, N. Dandekar, P. Mysore, et al., "Activity recognition from accelerometer data," in *AAAI-05*, Pittsburgh, USA, Ju. 2005, pp. 1541–1546.
[25] H. He, S. Chen, K. Li, et al., "Incremental learning from stream data," *IEEE Transactions on Neural Networks*, vol. 22, no. 12, pp. 1901–1914, Oct. 2011. doi: 10.1109/TNN.2011.2171713.
[26] H. Drucker, C. J. C. Burges, L. Kaufman, et al., "Support vector regression machines," *Advances in Neural Information Processing Systems*, vol. 9, pp. 155–161, 1997.

Manuscript received: 2015-07-20

Biographies

LIU Chengxuan (lcxstorm@163.com) is studying for the ME degree in the Department of Information and Communication Engineering, Shanghai Jiao Tong University, China. He received his BE degree from Shanghai Jiao Tong University. His research interests include pattern recognition, multisource navigation, and positioning technology.

DONG Zhenjiang (dong.zhenjiang@zte.com.cn) is the vice president of the Cloud Computing & IT Research Institute of ZTE Corporation. His main research areas are cloud computing, big data, new media, and mobile internet technologies.

XIE Siyuan (xie.siyuan7@zte.com.cn) is a pre-research engineer at ZTE Corporation. His main research areas are indoor positioning, IoT, and mobile internet technologies.

PEI Ling (ling.pei@sjtu.edu.cn) is an associate professor in the Department of Electronic Engineering, Shanghai Jiao Tong University, China. He received his PhD degree from Southeast University, China. His research interests include indoor and outdoor seamless positioning, context-aware technology, and motion pattern recognition.

ZTE Communications Guidelines for Authors

• Remit of Journal

ZTE Communications publishes original theoretical papers, research findings, and surveys on a broad range of communications topics, including communications and information system design, optical fiber and electro-optical engineering, microwave technology, radio wave propagation, antenna engineering, electromagnetics, signal and image processing, and power engineering. The journal is designed to be an integrated forum for university academics and industry researchers from around the world.

• Manuscript Preparation

Manuscripts must be typed in English and submitted electronically in MS Word (or compatible) format. The word length is approximately 3000 to 8000, and no more than 8 figures or tables should be included. Authors are requested to submit mathematical material and graphics in an editable format.

• Abstract and Keywords

Each manuscript must include an abstract of approximately 150 words written as a single paragraph. The abstract should not include mathematics or references and should not be repeated verbatim in the introduction. The abstract should be a self-contained overview of the aims, methods, experimental results, and significance of research outlined in the paper. Five carefully chosen keywords must be provided with the abstract.

• References

Manuscripts must be referenced at a level that conforms to international academic standards. All references must be numbered sequentially in-text and listed in corresponding order at the end of the paper. References that are not cited in-text should not be included in the reference list. References must be complete and formatted according to ZTE Communications Editorial Style. A minimum of 10 references should be provided. Footnotes should be avoided or kept to a minimum.

• Copyright and Declaration

Authors are responsible for obtaining permission to reproduce any material for which they do not hold copyright. Permission to reproduce any part of this publication for commercial use must be obtained in advance from the editorial office of *ZTE Communications*. Authors agree that a) the manuscript is a product of research conducted by themselves and the stated co-authors, b) the manuscript has not been published elsewhere in its submitted form, c) the manuscript is not currently being considered for publication elsewhere. If the paper is an adaptation of a speech or presentation, acknowledgement of this is required within the paper. The number of co-authors should not exceed five.

• Content and Structure

ZTE Communications seeks to publish original content that may build on existing literature in any field of communications. Authors should not dedicate a disproportionate amount of a paper to fundamental background, historical overviews, or chronologies that may be sufficiently dealt with by references. Authors are also requested to avoid the overuse of bullet points when structuring papers. The conclusion should include a commentary on the significance/future implications of the research as well as an overview of the material presented.

• Peer Review and Editing

All manuscripts will be subject to a two-stage anonymous peer review as well as copyediting, and formatting. Authors may be asked to revise parts of a manuscript prior to publication.

• Biographical Information

All authors are requested to provide a brief biography (approx. 100 words) that includes email address, educational background, career experience, research interests, awards, and publications.

• Acknowledgements and Funding

A manuscript based on funded research must clearly state the program name, funding body, and grant number. Individuals who contributed to the manuscript should be acknowledged in a brief statement.

• Address for Submission

magazine@zte.com.cn
12F Kaixuan Building, 329 Jinzhai Rd, Hefei 230061, P. R. China

ZTE COMMUNICATIONS

ZTE Communications has been indexed in the following databases:

- Cambridge Scientific Abstracts (CSA)
- China Science and Technology Journal Database
- Chinese Journal Fulltext Databases
- Inspec
- Ulrich's Periodicals Directory
- Wanfang Data—Digital Periodicals