Game-Theoretical Batch Identification of Invalid Signatures in Wireless Mobile Networks

Jing Chen, Quan Yuan, Guoliang Xue, Ruiying Du

Abstract—Digital signature has been widely employed in wireless mobile networks to ensure the authenticity of messages and identity of nodes. A paramount concern in signature verification is reducing the verification delay to ensure the network QoS. To address this issue, researchers have proposed the batch cryptography technology. However, most of the existing works focus on designing batch verification algorithms without sufficiently considering the impact of invalid signatures. The performance of batch verification could dramatically drop, if there are verification failures caused by invalid signatures. In this paper, we propose a Game-theory-based Batch Identification Model (GBIM) for wireless mobile networks, enabling nodes to find invalid signatures with the optimal delay under heterogeneous and dynamic attack scenarios. Specifically, we design an incomplete information game model between a verifier and its attackers, and prove the existence of Nash Equilibrium, to select the dominant algorithm for identifying invalid signatures. Moreover, we propose an auto-match protocol to optimize the identification algorithm selection, when the attack strategies can be estimated based on history information. Comprehensive simulation results demonstrate that GBIM can identify invalid signatures more efficiently than existing algorithms.

Index Terms—Batch identification, game theory, wireless mobile networks.

I. INTRODUCTION

Wireless Mobile Networks (WMNs) have brought significant convenience by enabling people to use applications on mobile devices (e.g., social media networks and electronic payment) [1]. However, due to their openness, such networks also provide opportunities to malicious nodes, who may threaten the network security by sending tampered or forged messages [2], [3]. To ensure the authenticity of messages and the identity of senders, one approach is to sign each outgoing message with a digital signature, and let the destinations verify the signature of each received message. Generally, signature verification induces extra delay and computational cost. Individual verification, the traditional way, could severely influence the Quality of Service (QoS) and the network availability, especially when there is intensive network traffic with massive signatures to verify, since it would result in unaffordable processing time and delivery delay.

Batch verification has been developed to improve the efficiency of signature verification, by processing a list of $n$ message-signature pairs as a batch [4]. It can quickly determine whether a batch has any invalid signatures. Specifically, batch verification methods return true if all of the $n$ signatures are valid, and false when there is at least one invalid signature. It can remarkably reduce the processing time compared to doing $n$ individual verifications [5], if there is no invalid signature in the messages. In reality, it is infeasible to assume that all signatures are valid. Adversaries can attempt to negate the advantages of batch verification by polluting signatures within a batch. Thus, batch identification, used to identify the invalid signatures in a batch, is required when the batch verification fails.

To efficiently identify the invalid signatures in bad batches, instead of verifying each signature individually, divide-and-conquer techniques have been proposed [6]. Those methods can dramatically reduce the identification time at different levels. However, there are two limitations in existing works. One is that many methods are designed only for some particular batch types, such as RSA-type batches [7], and pairing-based batches [8]. Though these works are state-of-the-art, it is challenging to apply them with the various batch verification algorithms. The other is that existing batch identification methods are usually suitable for a specific attack situation in terms of the ratio of invalid signatures. Their performance may heavily degrade if the ratio of invalid signatures varies when adversaries change attack frequencies and locations. In 2012, Akinyele et al. first proposed an automated tool for selecting the most suitable batch verification algorithm [9]. However, they did not sufficiently consider the automatic selection of batch identification. Therefore, designing a generic and auto-match batch identification solution towards the heterogeneous and dynamic attack scenario becomes significant.

In this paper, we propose a Game-theory-based Batch Identification Model (GBIM) in WMNs, enabling nodes to identify invalid signatures with the optimal delay under heterogeneous and dynamic attack scenario. We consider that a WMN consists of regular nodes and malicious nodes. Malicious nodes intend to interpose batch verification process by broadcasting false messages signed by invalid signatures with different frequencies, while regular nodes aim at finding the invalid signatures to eliminate the impact of malicious nodes, and to trace their identities. A game model is designed between a regular node and the malicious nodes around it, to find the optimal invalid signature identification strategy for the regular node to resist attacks. Our main contributions are summarized...
We design a game model to find the dominant invalid signature identification algorithm to resist various attack strategies. In addition, we prove the existence of Nash equilibrium between a regular node and the malicious nodes around it. Note that the three generic identification algorithms used as the members of strategy set are alternative, and our game model provides a paradigm to automatically select the dominant one to identify invalid signatures. As long as the invalid signature identification algorithms in the strategy set have their own advantages, they can be used with our model. Furthermore, GBIM can work for any signature scheme equipped with a batch.

Due to the dynamics of attack scenarios, the dominant strategy may not always be the optimal one. We propose an auto-match protocol to select the optimal strategy based on the dominant strategy and the history information.

The reminder of the paper is organized as follows. We review the related works in Section II. Section III introduces the network model and attack model. Three generic invalid signature identification algorithms are recast and analyzed in Section IV. Section V discusses the incomplete game model and Section VI presents our auto-match batch identification protocol. Finally, we conclude the paper in Section VIII.

II. RELATED WORK

Batch cryptography highlights a novel direction in computer and communication security. The concept of batch cryptography was introduced by Fiat in 1990 for an RSA-type signature [11], and the first efficient batch verifier was proposed by Naccache et al. in 1994 for DSA-type signatures [12]. Currently, researchers focus on two directions to apply the batch cryptography concept in WMNs: batch verification and batch identification. A batch verification algorithm is used to determine whether a set of signatures contain invalid ones. In 2008, considering that the verification of massive messages may induces huge time cost in mobile networks, Yu et al. proposed an efficient identity-based batch verification scheme to reduce the delay in network coding [13]. Zhang et al [14] discussed a batch signature verification scheme for the communications between vehicles and infrastructure to lower the total verification time. Horng et al. [15] presented a group signature and batch verification method for secure pseudonymous authentication in VANET. Even though those schemes could protect the authority of messages, their performance would be severely affected if there are invalid signatures existing in the verified batch.

On the other hand, batch identification is to find the bad signatures within a batch, when the batch verification fails. Existing batch identification algorithms have been developed into two main branches: special and generic. The special methods are designed for certain batch signature types such as RSA-type, DSA-type and pairing-type. Lee et al. [7] proposed a method to identify bad signatures in RSA-type batches. Later, Law and Matt [16] presented the quick binary and exponentiation method, to find invalid signatures in the pairing based signature schemes. Stanek [17] showed that method was flawed, and proposed an improved protocol to resist attacks. Matt [8] discussed a solution in pairing-based signature scheme, which can identify nontrivial numbers of invalid signatures in batches. The generic batch identification methods utilize the group testing technique to find invalid signatures with the minimal number of tests, which can be applied with any signature types. Pastuszak et al. designed a divide-and-conquer verifier [6], which split a batch instance into sub ones, and applied the generic test to each sub-batch recursively, until all bad signatures are identified. Zaverucha et al [18] presented and compared some group testing algorithms for finding invalid signatures. Zhang et al. [10] adopted the group testing technique to find invalid signatures in a batch in mobile networks. Lee et al. [19] proposed a secure batch verification with group testing to improve the real-time performance of mobile networks.

However, those generic methods are usually suitable for a specific attack situation in terms of the ratio of invalid signatures. When adversaries change attack frequencies and locations, and consequently the ratio of invalid signatures varies, their performance may heavily degrade. For that, we propose a generic and auto-match batch identification solution, which can find the optimal algorithm to identify invalid signatures in the heterogeneous and dynamic attack scenario for WMNs.

III. SYSTEM MODEL

A. Network Model

We consider that the network has two layers as shown in Figure 1. The bottom layer consists of mobile nodes accessing the network via bluetooth, wifi, GSM, 3G, etc. Each node has its own public/private keys, which are used to sign the outgoing messages and to verify the signatures of the received messages. The top layer is composed of an authority center and base stations. The authority center manages the operations of all legal nodes’ keys in network, including generation, distribution, storage, update and destruction. If mobile nodes directly communicate with each other by wifi or bluetooth, they should mutually verify the validity of the other party. Otherwise, base stations forward their messages, and have to verify the validity of requests. Hence, both base stations and mobile nodes can be the targets of attackers. And they should protect their own security, and identify the invalid signatures in the false messages by themselves.
B. Attack Model

We assume the network consists of regular nodes (called verifiers), and malicious nodes (called attackers), which are the two players in the game. One verifier may have multiple malicious neighbors round it as attackers. For a verifier, its attackers intend to interpose its batch verification process by broadcasting false messages with invalid signatures, while the verifier needs to identify the invalid signatures quickly to resist that attack. In this paper, the verifier can be a base station or a mobile node. In addition, in our attack model, for a verifier, its attackers have the following preferences.

- Attackers cannot interfere or control the key pair generation and assignment, since those can be conducted by secure channels or offline methods. They launch attacks in the way of broadcasting false messages with invalid signatures, to disturb the batch verification process, and to consume the verifier’s resources.
- Attackers may have multiple attack strategies. To maximize the attack effect, attackers can change their attack strategies occasionally. Also, multiple attackers can attack one single verifier simultaneously with different attack strategies.
- Attackers are divided into different types based on their preferences. For example, some attackers consider the risk of being traced, but others do not have such concerns.
- Attackers can acquire the public information of the verifier, such as the public key and the cryptographic algorithm.

IV. GENERIC IDENTIFICATION ALGORITHMS AND ANALYSIS

Generic invalid signature identification algorithms for a bad batch usually adopt the group testing technology. In this section, we describe and analyze the idea of three generic algorithms based on the representative group testing technologies, including individual identification, generalized binary splitting (GBS), Li’s s-stage [10], [20], to identify d invalid signatures in a batch of n messages.

Algorithm 1 Condensed Binary Identification Algorithm

1: while true do
2:   if $n \leq 2d - 2$ then
3:     Verify $n$ messages using II;
4:     return;
5:   else
6:     $z = n - d + 1$;
7:     $\theta = \lceil \log(z/d) \rceil$;
8:     Verify the prevenient $2^\theta$ messages with batch verification;
9:     if verification succeeds then
10:        $n = n - 2^\theta$;
11:    continue;
12:   else
13:        identify an invalid signature by basic binary identification after verifying $v$ messages;
14:        $n = n - 1 - v$;
15:        $d = d - 1$;
16:    continue;

A. Individual Identification (II)

One simple solution to identify all invalid signatures in a bad batch, is verifying each signature individually. Note that signatures are not aggregated with others until all invalid signatures have been found. Many batch verification schemes, which mainly focus on the batch verification process, adopt this algorithm. That is, once the batch verification fails, Individual Identification (II) is employed to find all the invalid signatures. Obviously, the time complexity of II is $O(n)$.

B. Condensed Binary Identification (CBI)

Inspired by the basic binary identification algorithm in [9], we present an improved scheme called the Condensed Binary Identification (CBI) algorithm. In the basic binary identification, it first divides the $n$ messages into two groups of the same size. Then, those two groups are verified using batch verification individually. If the batch verification succeeds, it means that there is no invalid signature in that group. Otherwise, messages in that group will be further divided into two sub-groups, and each sub-group is verified individually. That process repeats until all of the messages pass the batch verification. CBI improves the basic binary identification by adjusting the group size for efficiency. Concerning the probability, the ideal situation is that, each sub-group of $\lceil n/d \rceil$ messages has one invalid signature, where $\lceil n/d \rceil$ denotes the largest integer not greater than $n/d$. If we can adjust the sub-group size based on the number of the remaining invalid signatures, it can reduce the reverification times in attacks. CBI is described as Algorithm 1, where $z$, $\theta$ and $v$ are three intermediate variables. The time complexity of CBI is $O(d \log(n/d))$ [21].

C. Multiple Rounds Identification (MRI)

In Multiple Rounds Identification (MRI) algorithm, we identify the invalid signatures in an iterative way which has
Algorithm 2 Multiple Rounds Identification Algorithm

1: Copy \( n \) sample messages to test\_batch;
2: while \( i \leq m \) do
3: \( \gamma_i = \left\lfloor \frac{n/d}{m} \right\rfloor \);
4: \( \delta_i = \left\lfloor \frac{n}{\gamma_i} \right\rfloor + 1 \);
5: Divide test\_batch into \( \delta_i \) groups of \( \gamma_i \) messages (may be less than \( \gamma_i \) in the last group);
6: for \( j = 0 \) to \( j < \delta_i \) do
7: if Batch verification on group \( j \) succeeds then
8: Remove the contents of group \( j \) from test\_batch;
9: \( j++ \);
10: \( i = i + 1 \);
11: return test\_batch;

Table I

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual Identification (II)</td>
<td>( O(n) )</td>
</tr>
<tr>
<td>Condensed Binary Identification (CBI)</td>
<td>( O(d \log(n/d)) )</td>
</tr>
<tr>
<td>Multiple Rounds Identification (MRI)</td>
<td>( O(\log(n/d)) )</td>
</tr>
</tbody>
</table>

\( m \) (\( 2 \leq m \leq n \)) rounds, as described in Algorithm 2. In the first round, the \( n \) pending messages are divided into \( \delta_1 \) groups, and each group has \( \gamma_1 \) messages except the last group. Then, each group is verified respectively. The groups identified with invalid signatures are aggregated as a new pending message batch. In the second round, that new message batch is divided into \( \delta_2 \) groups of \( \gamma_2 \) messages. In general, in round \( i \), \( 2 < i < m \), messages from the contaminated groups of round \( i-1 \) are pooled, and arbitrarily divided into \( \delta_i \) groups of \( \gamma_i \) (some possibly less than \( \gamma_i \)) size. A batch verification test is performed on each group. Note that \( \gamma_m \) is set to be 1. Thus every invalid signature is identified at round \( m \). The time complexity of MRI is \( O(\log(n/d)) \) \cite{10, 15}.

D. Performance Comparison

Based on the above analysis, we summarize the time complexity of these algorithms in Table I. In addition, in Figure 2, we investigate the relationship between the number of invalid signatures and the number of required batch verification tests, when the number of messages \( n \) varies. Figure 2(a)-(d) presents the situation where \( n \) is equal to 100, 150, 200, and 250, respectively. The result shows that given a specific \( n \), the number of required batch verifications ascends as the number of invalid signatures upper-bound increases in CBI and MRI, but not in II. Also, CBI has a lower start point and a larger slope, while MRI has a higher start point, and its slope turns smaller. As a result, in Figure 2, CBI and MRI eventually meet at a point, marked as point 1, while II and CBI meet at another point marked as point 2. For our game model design, we define that point 1 as the demarcation point of attack strategy, because the optimal invalid signature identification algorithm is changed at point 1. That is, if the number of invalid signatures is less than that of point 1, given the message number \( n \), it means that attackers adopt the low-frequency attack, denoted by strategy \( L \). Otherwise, they employ the high-frequency attack, represented by strategy \( H \). Each invalid signature identification algorithm has its own advantage under a specific attack strategy. If we can automatically choose the invalid signature identification algorithms based on the attack strategy, it can achieve better performance.

V. BATCH IDENTIFICATION GAME MODEL

Algorithms in Section IV assist verifiers to find invalid signatures when the batch verification fails. From Figure 2, we see that it is challenging to find an optimal algorithm for a general purpose, since those algorithms have different capabilities under different attack strategies. Therefore, the first step of our scheme is to find the dominant strategy by the game model from the three algorithms in Section IV.

A. Game Model Definition

We consider the problem between a verifier and its attackers as a dynamic game, where attackers select the attack strategy first, and the verifier picks the invalid signature identification algorithm accordingly. The definition of GBIM is represented by a triple \( (PL, S, U) \), where \( PL \) is the player set, \( S \) denotes the strategy set of players, and \( U \) stands for the payoff function set. The detailed description is as follows.

1) Players: The player set is represented by \( PL = \{PL_i\}_{i=1}^\infty \), where \( i \) is the index number of the player, and \( l \) is the total number of players. Obviously, set \( PL \) includes two players (\( l = 2 \)). One is the verifier, and the other is the attackers, which can be the verifier’s malicious neighbors.

2) Strategy set: The strategy set of players is \( S = \{S_i\}_{i=1}^\infty \). Different players in the game may have different strategies. For attackers, the adopted strategies fall into two types, high-frequency attack \( H \) and low-frequency attack \( L \), in terms of the ratio of invalid signatures. Hence, the strategy set

![Figure 2](image_url)
of attackers is denoted as \( S_{\text{attack}} = \{H, L\} \). On the other side, the verifier’s strategy set is \( S_{\text{ver}} = \{\text{CBI, MRI, II}\} \), which includes the three identification algorithms defined in Section IV.

3) Payoff function: Each regular node acts as a verifier to protect its QoS. Let \( Q \) denote the communication benefit in an ideal network environment. For the verifier \( V \), the payoff function is \( u_V = b_V - c_V \), where \( b_V \) is the communication benefit \( Q \), and \( c_V \) indicates the total cost of batch verification and invalid signature identification. The cost of batch verification for \( n \) messages, denoted as \( C_{BV}^n \), is determined by the batch verification algorithm. And the cost of invalid signature identification algorithm is represented by \( \alpha(j, k) \), which is determined by the identification strategy \( j \in \{\text{CBI, MRI, II}\} \), and the attack strategy \( k \in \{H, L\} \). To simplify notations, we use 1, 2, 3 to index the algorithms CBI, MRI and II. Note that \( \alpha(j, k) \) is determined by the number of required batch verification tests. With the above discussion, the payoff function of verifier \( V \) can be defined as

\[
u_V = Q - C_{BV}^n - \alpha(j, k).
\]

Recall that the intention of attackers is to consume the verifier’s resources by broadcasting false messages, and eventually downgrade the QoS of the network. The payoff function of attackers \( A \) is \( u_A = b_A - c_A \), where \( b_A \) is the loss of QoS, which is affected by the verification cost of the verifier. Therefore \( b_A = C_{BV}^n + \alpha(j, k) \), \( c_A \) indicates the attack cost, which is determined by the number of the broadcasted false messages with invalid signatures, denoted by \( \sigma(k) \) \( (k \in \{H, L\}) \). Therefore, the payoff function is

\[
u_A = C_{BV}^n + \alpha(j, k) - \sigma(k).
\]

B. Incomplete Game Scenarios

Generally, attackers hiding in the darkness can monitor and collect the verifier’s information as common knowledge. But it is hard for the verifier to acquire the complete information of attackers in advance. Recall that there are different types of attackers with various preferences in the network, and each attacker has several attack strategies. The verifier does not exactly know which strategies are adopted by which type of attackers. Our Game-theory based Batch Identification Model (GBIM) is designed toward that scenario, which is an incomplete information game.

Specifically, we divide attackers into two types based on their preferences. One type is hot-headed, who does not consider the possibility of traceback by the verifier, while the other is more cautious, who does some extra work such as utilizing zombie [22] to confuse the verifier in order to protect its identity. The type of attackers can be identified by the Wireless Intrusion Prevention System (WIPS) according to their behaviors, which is out of our scope. We denote the cost of extra works for anti-tracking as \( \beta(k) \) \( (k \in \{H, L\}) \), and it grows as the number of false messages increases.

For hot-headed attackers, the benefit of strategy \( H \) is greater than that of strategy \( L \). Hence, they are inclined to employ strategy \( H \). We can get \( \alpha(x, H) - \sigma(H) > \alpha(x, L) - \sigma(L) \), where \( x \) is the index of the invalid signature identification algorithm. For cautious attackers, the extra confusion work will protect itself, and that work is more effective in strategy \( L \). Hence, we can achieve that \( \alpha(x, H) - \sigma(H) - \beta(H) > \alpha(x, L) - \sigma(L) - \beta(L) \). Since the cost of the verifier is a common knowledge, the verifier only has one type. And the payoff array of hot-headed attacks is as Table II and that of cautious attacks is shown in Table III.

Because both types of attackers may exist simultaneously, we use \( P \) to denote the probability that attackers are hot-headed, and correspondingly, the probability of cautious attackers is \( 1 - P \). To analyze the incomplete information game, we design the game tree of our model as shown in Figure 3. From Figure 3, we find that GBIM follows Theorem V.1.

**Theorem V.1. GBIM has at least one Nash equilibrium.**

**Proof:** Let us analyze the two different cases of our game model respectively. In the discussion below, \( E_{uv}(j) \) represents the benefit of the verifier \( V \) when it chooses the identification algorithm \( j \), and \( E_{uv}(k) \) indicates the benefit of attackers \( A \) when they choose the attack strategy \( k \).

**Case I:** Two type attackers adopt the same attack strategy.

a) Both take strategy \( H \). In this situation, the benefit of the verifier is as follows for algorithms CBI, MRI and II.

\[
E_{uv}(\text{CBI}) = Q - C_{BV}^n - \alpha(1, H)
\]

\[
E_{uv}(\text{MRI}) = Q - C_{BV}^n - \alpha(2, H)
\]

\[
E_{uv}(\text{II}) = Q - C_{BV}^n - \alpha(3, H)
\]

From the analysis in Section IV, we know that \( E_{uv}(\text{MRI}) \) is the largest, and MRI is the dominant algorithm among those three algorithms in this situation. Then, let us compare the
And it requires, 

\[ E_{u_A}(H) = P(C_{BV}^0 + \alpha(2, H) - \sigma(H)) \]

\[ + (1 - P)(C_{BV}^0 + \alpha(2, H) - \sigma(H) - \beta(H)) \]

\[ E_{u_A}(L) = P(C_{BV}^0 + \alpha(2, L) - \sigma(L)) \]

\[ + (1 - P)(C_{BV}^0 + \alpha(2, L) - \sigma(L) - \beta(L)) \]

If the attackers’ benefit with strategy \( H \) is greater than that with strategy \( L \), \( E_{u_A}(H) > E_{u_A}(L) \), it requires,

\[ 0 < P < \frac{\alpha(2, H) - \alpha(2, L) + \sigma(L) - \sigma(H)}{\beta(H) + \beta(L)} + 1 \]  

(1)

From the Section IV, we have \( \alpha(2, H) - \sigma(H) > \alpha(2, L) - \sigma(L) \). Thus, the Equation 1 is always satisfied.

Hence, (Attacker: hot-headed attacker, H, cautious attacker, H, P; Verifier: MRI) is a candidate Nash Equilibrium.

b) Both adopt strategy \( L \). Similar to the analysis of strategy \( H \), we see that CBI is the dominant algorithm in this situation. And it requires,

\[ \frac{\alpha(1, H) - \alpha(1, L) + \sigma(L) - \sigma(H)}{\beta(H) + \beta(L)} + 1 < P < 1 \] 

(2)

However, from the Section IV, since \( \alpha(1, H) - \sigma(H) > \alpha(1, L) - \sigma(L) \), Equation 2 does not stand.

Hence, (Attacker: hot-headed attacker, L, cautious attacker, L, P; Verifier: CBI) is not a Nash Equilibrium.

In summary, GBIM only exists one candidate Nash equilibrium in this case.

Case 2: Two type attackers adopt different strategies.

In this case, the benefit of the verifier is displayed as follows for algorithms CBI, MRI and II.

\[ E_{uv}(CBI) = P(Q - C_{BV}^0 - \alpha(1, H)) \]

\[ + (1 - P)(Q - C_{BV}^0 - \alpha(1, L)) \]

\[ E_{uv}(MRI) = P(A - C_{BV}^0 - \alpha(2, H)) \]

\[ + (1 - P)(Q - C_{BV}^0 - \alpha(2, L)) \]

\[ E_{uv}(II) = P(Q - C_{BV}^0 - \alpha(3, H)) \]

\[ + (1 - P)(Q - C_{BV}^0 - \alpha(3, L)) \]

First of all, let us analyze the situation that CBI is the dominant algorithm. Then, we have \( E_{uv}(CBI) > E_{uv}(MRI) \) and \( E_{uv}(CBI) > E_{uv}(II) \).

And we get,

\[ 0 < P < \frac{\alpha(2, L) - \alpha(1, L)}{\alpha(2, L) - \alpha(1, L) + \alpha(1, H) - \alpha(2, H)} \]  

(3)

and

\[ 0 < P < \frac{\alpha(3, L) - \alpha(1, L)}{\alpha(3, L) - \alpha(1, L) + \alpha(1, H) - \alpha(3, H)} \]  

(4)

For attackers, due to \( \alpha(x, H) - \sigma(H) > \alpha(x, L) - \sigma(L) \), the hot-headed attackers select strategy \( H \). On the other hand, since \( \alpha(x, H) - \sigma(H) - \beta(H) < \alpha(x, L) - \sigma(L) - \beta(L) \), the cautious attackers pick strategy \( L \). Hence, (Attacker: hot-headed, H, cautious, L, P; Verifier: CBI) is a candidate Nash Equilibrium as long as \( P \) satisfies Equation 3 and Equation 4.

Similarly, (Attacker: hot-headed, H, cautious, L, P; Verifier: MRI) and (Attacker: hot-headed, H, cautious, L, P; Verifier: II) also are candidate Nash Equilibriums when \( P \) has the proper value. The relationship between Nash Equilibrium and the value range of \( P \) is summarized in Table IV.

Note that the cautious attackers gain more benefit in the Nash Equilibriums of Case 2 than that of Case 1. Thus, in practice, the cautious attackers are more inclined to adopt low frequency attack strategy. In another word, if \( P \) is in the appropriate range, the Nash Equilibrium of Case 2 is the prime choice of the verifier. However, from Table IV, we find that the Nash Equilibrium may not exist if the ratio of hot-headed attackers \( P \) is in some special range. When that situation happens, the Nash Equilibrium in Case 1 will be used.

Hence, GBIM at least has one Nash Equilibrium. □

With the game model, the verifier can pick the dominant algorithm to identify valid signatures. For example, assuming the ratio of hot-headed attackers \( P \) is 50%,

<table>
<thead>
<tr>
<th>Nash Equilibrium</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Attacker: hot-headed, H, cautious, L, P; Verifier: CBI)</td>
<td>( 0 &lt; P &lt; \frac{\alpha(2, L) - \alpha(1, L)}{\alpha(2, L) - \alpha(1, L) + \alpha(1, H) - \alpha(2, H)} ) and ( 0 &lt; P &lt; \frac{\alpha(3, L) - \alpha(1, L)}{\alpha(3, L) - \alpha(1, L) + \alpha(1, H) - \alpha(3, H)} )</td>
</tr>
<tr>
<td>(Attacker: hot-headed, H, cautious, L, P; Verifier: MRI)</td>
<td>( \frac{\alpha(2, L) - \alpha(1, L)}{\alpha(2, L) - \alpha(1, L) + \alpha(1, H) - \alpha(2, H)} &lt; P &lt; 1 )</td>
</tr>
<tr>
<td>(Attacker: hot-headed, H, cautious, L, P; Verifier: II)</td>
<td>( \frac{\alpha(3, L) - \alpha(1, L)}{\alpha(3, L) - \alpha(1, L) + \alpha(1, H) - \alpha(3, H)} &lt; P &lt; 1 )</td>
</tr>
</tbody>
</table>
Nash Equilibrium in Case 1, and adopt MRI as the dominant in Case 2. Hence, the verifier will find the strategy from the 
Verifier: II)
previously, when 
\( P_{\text{yes}} \) picks the strategy as the dominant choice accordingly.

achieve a Nash Equilibrium from the results of Case 2. If 

The initialization phase aims at negotiating and updating some shared security information for batch identification. The information includes several public parameters, such as signature algorithm, hash function, public key, etc., as well as a few private parameters, such as private key and identity information. The initialization operation involves two cases. One is that when a new regular node joins the WMN, the authority server distributes the related security information to it offline. The other is that when a regular node leaves the WMN, the authority server updates and broadcasts that information to other regular nodes with signatures. Note that in this paper, we focus on identifying invalid signatures. The schemes of key distribution and management are out of our scope.

A. Initialization Phase

The initialization phase consists of configuring system parameters to resist attacks; (b) Decision phase, where the verifier estimates the existence of invalid signatures, and selects the optimal invalid signature identification algorithm. The process of protocol is illustrated as Figure 4.

B. Decision phase

After initialization, mobile nodes could send messages with signatures, and verify the signatures of the received messages. To protect the security of the network, signature verification must be conducted periodically. Hence, we divide time into multiple periods, and the verifier must automatically decide which invalid signature identification algorithm to apply in each period. Within a period, the verifier first tests the messages with a batch verification algorithm. If it succeeds, then nothing is done until the next period comes. Otherwise, the verifier must find the invalid signatures using a batch identification algorithm. Secondly, we notice that the flow rate may change quickly, but the attack strategies may not change that frequently. Therefore, if the attack strategy is not changed in previous two periods, the verifier can estimate attackers’ current attack strategy based on the observed behavior history of attackers, and select the proper algorithm accordingly. Otherwise, the game model can provide the dominant choice for batch identification. Note that in the latter case, the verifier cannot estimate the attackers’ strategies at all. Hence, before applying the game model to find the optimal algorithm to identify invalid signatures, the verifier must check whether it can estimate the current attack strategy based on history information. Thirdly, no matter which identification algorithm is chosen, the verifier needs to collect the number of invalid signatures, and analyze the current attack strategy. Such information will be stored as history data for next period. Finally, the verifier also records the possibility of different type attackers as the accessorial data for the incomplete game. The pseudo-code is shown in Algorithm 3, where \( S_h^{\text{B}}(V) \) indicates the batch identification strategy of the verifier \( V \) in period \( h \). \( P_h \) denotes the ratio of hot-headed attackers in the \( h \)th period of our batch identification protocol.

VI. AUTO-MATCH PROTOCOL FOR BATCH IDENTIFICATION

With the previous discussion, we find that the Nash Equilibriums exist in the incomplete game scenario, which provides the dominant choice for the verifier to get the maximum benefit. However, that works based on the assumption that the verifier cannot acquire sufficient accessorial data, such as the recent number of the invalid signatures. If the verifier can observe that the invalid signatures are few during some period, MRI algorithm, which might be the choice from Nash Equilibrium, obviously is not the optimal one. Therefore, the Nash Equilibrium is only useful when the verifier does not have any accessorial data, or attackers frequently change their attack strategies.

In this section, we formally propose an auto-match protocol for batch identification, which can automatically choose the optimal identification algorithm based on both the game theory analysis and the observed recent attackers’ behaviors. Our batch identification protocol can be implemented within two phases: (a) Initialization phase, where the verifier configures system parameters to resist attacks; (b) Decision phase, where the verifier estimates the existence of invalid signatures, and selects the optimal invalid signature identification algorithm. The process of protocol is illustrated as Figure 4.
Algorithm 3 Decision phase

1: while true do
2: if (Batch verification succeeds) then
3: \( h = h + 1 \);
4: continue;
5: else
6: if \( (S^b_{V} - 2 = S^b_{V} - 1) \) then
7: if \( (S^c_{V} - 1 = L) \) then
8: Select CBI algorithm;
9: else
10: Select MRI algorithm;
11: else
12: Select the invalid signature identification algorithm with GBIM;
13: Record \( S^c_{V} \) according to the results of invalid signature identification;
14: Record \( P_h \) as the accessorial data;
15: \( h = h + 1 \);
16: if (Process should be ended) then
17: Break;

VII. SIMULATION

A. Simulation setting

In our simulation, we define and evaluate the time cost of cryptographic operations required in the batch identification. We adopt the experiment methodology in [23], which observes the processing time for a curve of embedding degree \( k = 6 \) and 160-bit \( q \), running on an Intel Pentium IV 3.0GHZ machine. Also, as we mentioned in Section VI-B, the batch identification is conducted every 5 seconds as one period. The other simulation parameters are presented in Table V.

We choose ECDSA [24] as the batch verification algorithm, which belongs to IEEE1609.2 standard. Note that our scheme is independent from batch verification algorithms. Therefore, if we utilize any other batch verification algorithms, even though the delay may be different, the relationship of the compared algorithms should be the same. According to the IEEE1609.2 standard, verifying a single signature requires 4 MapToPoint hash operations, and each operation needs 0.6ms. Hence, the total delay of verifying a single signature is 2.4ms.

B. Results

In this section, we consider evaluating different delays with different ratio of attackers, ratio of hot-headed attackers and ratio of cautious attackers. The ratio of attackers is the ratio of the number of malicious nodes to that of total nodes in simulation. The percent of hot-headed and cautious attackers in malicious nodes are called ratio of hot-headed attackers and the ratio of cautious attackers, respectively. To facilitate comparison, we only consider two attack strategies: strategy \( L \) with 15% invalid signatures, and strategy \( H \) with 50% invalid signatures. Figure 5(a), 5(b), and 5(c) display the relationship between the identification delay and the elapsed time, where the ratios of attackers are 30%, 40% and 50%, individually. Furthermore, the ratios of cautious attackers are 70%, 50% and 20%, respectively.

Figure 6(a), 6(b), and 6(c) indicate the total delays of algorithms when time elapses, where the ratios of attackers are 30%, 40% and 50%, respectively. And the ratios of cautious attackers are 70%, 50% and 20%, individually.

The simulation results are summarized below:

- It takes several periods for GBIM to reach the Nash Equilibrium, since it is hard for verifiers to acquire the precise attack information at once. In Figure 5, the identification delay decreases at beginning, and finally turns stable at some level. Besides, as the ratio of attackers and the ratio of hot-headed attackers increases, the time to reach the
Nash Equilibrium becomes larger, and the stable value of identification delay also rises.

- GBIM is a self-optimizing algorithm. No matter what ratio of attackers is, the total delay of each algorithm increases as time goes by. The growth of other three algorithms’ delay is close to linear, while the slope of GBIM decreases continually, and reaches a stable status after some point. It means that GBIM can optimize the delay with time elapsing.
- GBIM has the best overall performance in those four protocols. When the ratio of attackers is 30%, the relationship of identification delay in each algorithm is $II > MRI > CBI > GBIM$. Also, we see that the relationship is $II > CBI > MRI > GBIM$, when the ratio of attackers turns 40%, and $CBI > II > MRI > GBIM$, when the ratio of attackers becomes 50%. Moreover, we observe that CBI does better when the number of attacks is relatively small, and MRI does better when the number of attacks grows. GBIM can select the optimal algorithm to identify invalid signatures under different attack strategies.

VIII. CONCLUSION

To identify invalid signatures in batches with the optimal delay, we propose a game-theory based batch identification model, named GBIM, which consists of three components. We first analyze the performance of three invalid signature identification algorithms, and discuss their advantages under different attack strategies. Then, we design a game model with those three algorithms in the incomplete game scenario, and prove the existence of Nash Equilibrium. Finally, we discuss an auto-match protocol to improve the practicability of our game model, considering the possibility that verifiers may estimate attackers’ strategies. We prove that our protocol can choose the optimal algorithm for batch identification, under heterogeneous and dynamic attack scenario in WMNs.

REFERENCES


