รายงานวิจัยฉบับสมบูรณ์

โครงการ การปกป้องเครือข่ายอุปกรณ์ส่งสัญญาณด้วยศาสตร์แห่งการสื่อสารอย่างปลอดภัย

(Securing Sensor Networks with Cryptography)

โดย ดร. ชนาทิพย์ นามเปรมปรีดิ์ และคณะ

พฤศจิกายน 2551

สัญญาเลขที่ MRG5080074

รายงานวิจัยฉบับสมบูรณ์

โครงการ การปกป้องเครือข่ายอุปกรณ์ส่งสัญญาณด้วยศาสตร์แห่งการสื่อสารอย่างปลอดภัย

(Securing Sensor Networks with Cryptography)

คณะผู้วิจัย	สังกัด
1. ดร. ชนาทิพย์ นามเปรมปรีดิ์	ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยธรรมศาสตร์
2. Prof. Dr. Mihir Bellare	Department of Computer Science and Engineering, University of California, San Diego, USA

สนับสนุนโดยทบวงมหาวิทยาลัย และสำนักงานกองทุนสนับสนุนการวิจัย
(ความเห็นในรายงานนี้เป็นของผู้วิจัย ทบวงฯ และสกว. ไม่จำเป็นต้องเห็นด้วยเสมอไป)

1. บทคัดย่อ

ในปัจจุบันเครือข่ายอุปกรณ์ส่งสัญญาณได้รับความสนใจเป็นอย่างมาก เนื่องจากหน่วยประมวลผลขนาดเล็กมีราคาต่ำลง จึงมีการติดตั้งหน่วยประมวลผลบนอุปกรณ์ส่งสัญญาณชนิดต่าง ๆ เพื่อให้อุปกรณ์เหล่านี้ทำงานร่วมกันได้ในระบบต่าง ๆ เช่น ระบบควบคุมสัญญาณจราจร ระบบเฝ้าสังเกตสภาวะแวดล้อม ระบบพยากรณ์อากาศ ฯลฯ ลักษณะสำคัญประการหนึ่งของหลายระบบที่มีการนำอุปกรณ์ส่งสัญญาณมาใช้คือ ระบบเหล่านี้จะมีการกระจายอุปกรณ์ส่งสัญญาณไว้ในพื้นที่ที่สนใจศึกษาเพื่อให้อุปกรณ์เหล่านี้เก็บข้อมูลต่าง ๆ เพื่อนำไปทำการวิเคราะห์ต่อไป เป็นที่แน่นอนว่า ในระบบที่มีภารกิจที่สำคัญนั้น ควรมีการยืนยันความน่าเชื่อถือของข้อมูลเหล่านี้ (เช่น ในระบบที่ใช้ในทางการทหาร หรือในระบบวัดสภาวะแวดล้อมในโรงงานปฏิกรณ์ปรมาณู) การปกป้องข้อมูลเหล่านี้ จำเป็นต้องพึ่งพาเทคโนโลยีในทางศาสตร์แห่งการสื่อสารอย่างปลอดภัยเป็นอย่างยิ่ง ความท้าทายที่สำคัญอย่างหนึ่งคือ การนำเทคโนโลยีเหล่านี้มาปรับใช้ในระบบที่มีการใช้อุปกรณ์ส่งสัญญาณที่มีหน่วยประมวลผลขนาดเล็กและมีความสามารถจำกัด ในโครงการนี้ คณะผู้วิจัยได้ทำการออกแบบวิธีการและโปรโตคอลที่สามารถนำมาใช้ในระบบอุปกรณ์ส่งสัญญาณได้ วิธีการและโปรโตคอลที่ออกแบบได้แก่ รหัสยืนยันความแท้จริงของข้อมูลแบบนิรนาม ระบบแคพช่า และระบบลายเซ็นอิเล็กทรอนิกส์แบบผลรวม โดยทั้งสามสิ่งนี้ มีการทำงานที่มีประสิทธิภาพ อีกทั้งทางคณะผู้วิจัยยังได้เขียนข้อพิสูจน์ทางคณิตศาสตร์ เพื่อยืนยันว่าระบบเหล่านี้ให้ความปลอดภัยในการทำงานได้จริงอีกด้วย

คำสำคัญ: ศาสตร์แห่งการสื่อสารอย่างปลอดภัย การพิสูจน์ความปลอดภัย รหัสยืนยันความแท้จริงของข้อมูล แคพช่า ลายเซ็นอิเล็กทรอนิกส์ อุปกรณ์ส่งสัญญาณ

Abstract

There has been much interest in sensor network applications in recent years. With decreasing prices of processors as one of the major driving forces, many applications now equip sensors with processors and let them cooperate to achieve a common goal. Applications of this type include traffic control, eco-system monitoring, and forecasting systems. The most common applications involve scattering sensors in some environment so that they can gather data for further analysis. Clearly, in mission-critical applications, it is undesirable to gather data that are unreliable or potentially-corrupt (maliciously or otherwise). The need to protect information communicated among the sensors can only be addressed by cryptography. The challenge is to adapt cryptographic technology to the special needs of sensor network applications. Under this grant, we have designed schemes and protocols that can be applied to sensor network applications. The schemes and protocols we investigated are blind message authentication codes, CAPTCHAs (Completely Automated Public Turing tests to tell Computers and Humans Apart), and aggregate signatures. The proposed schemes and protocols are not only efficient but also provably secure.

Keywords: cryptography, security proofs, message authentication codes, CAPTCHAs, digital signatures, sensor networks

2. Executive Summary

2.1 ชื่อโครงการ

การปกป้องเครือข่ายอุปกรณ์ส่งสัญญาณด้วยศาสตร์แห่งการสื่อสารอย่างปลอดภัย
Securing sensor networks with cryptography

2.2 ชื่อหัวหน้าโครงการ

ผศ.ดร. ชนาทิพย์ นามเปรมปรีดิ์
ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์
มหาวิทยาลัยธรรมศาสตร์ ศูนย์รังสิต
อ. คลองหลวง จ. ปทุมธานี 12120
Phone: +662 564–3001–9 ต่อ 3063
FAX: +662 564–3010, +662 564–3001–9 ต่อ 3037
E-mail: meaw@alum.mit.edu

2.3 สาขาวิชาที่ทำงานวิจัย

Cryptography, Computer and network security, Distributed systems

2.5 ระยะเวลาดำเนินงาน

2 ปี

2.6 ปัญหาที่ทำการวิจัย และ ความสำคัญของปัญหา

Over the past decade, the Internet has become an important medium for digital communication. Few technologies have seen such exponential growth in popularity. Naturally, as people, businesses, and governments become reliant upon the Internet and digital communication technologies in general, there has been an ever-growing concern over the security issues involved in the communication. Cryptography is a field that addresses these concerns. Over the years, modern cryptography has produced a myriad of tools for secure communication, and many software packages have been written based on these tools and are in prevalent use today.

More recently, however, a new type of network interconnection has emerged, namely the *sensor network*. With decreasing prices of processors as one of the major driving forces, many applications now equip sensors with processors and let them cooperate to achieve a common goal. Applications of this type include traffic control, eco-system monitoring, and forecasting systems. The most common applications involve scattering sensors in some environment so that they can gather data for further analysis. As a concrete example, in civil engineering, sensors can be used to gather data in order to monitor current conditions of a structure such as a bridge or a building. For disaster prevention, sensors can be used to monitor activities in the ocean to detect disturbances or abnormal behaviors in order to issue warnings of a pending disaster. For

the military, sensors may be used for surveillance by scattering them about in a volatile area to monitor suspicious activities.

Clearly, in mission-critical applications, it is undesirable to gather data that are unreliable or potentially-corrupt (maliciously or otherwise). Consider sensor networks in which devices gather sensitive information such as temperatures in a nuclear reactor or voltage levels on a power line. Each device sends the information to a data-gathering node which may later pass it along to others for further analysis or simply file it on some permanent storage device. In these systems, it may be necessary to prevent tampering and/or forgery of data being gathered so as to prevent malicious attackers from, say, triggering a false alarm indicating that the reactor or the power grid are malfunctioning, or worse, preventing a true alarm from being noticed. As part of a solution to this problem, each device may use a cryptographic construct, namely digital signatures, to ensure the authenticity of every piece of the data that it sends to a data-gathering node.

This need, i.e., the need to protect information communicated among the sensors, can only be addressed by cryptography. The challenge is to adapt cryptographic technology to the special needs of sensor network applications. Current cryptographic tools are designed for applications on general purpose computers, not tiny processors on low-powered sensors. The unique characteristics of sensor networks are limitations in processing power, strict requirements in power consumption, and the typical large number of communicating nodes involved. These factors pose a challenge for system designers to ensure that the nodes communicate securely while meeting operational requirements of the applications.

There are a myriad of security goals in any particular sensor network application, e.g. to route messages among the sensors securely, to ensure integrity and secrecy of messages, to prevent compromised nodes from interfering with the operation of the network as a whole, and to do all of the above efficiently.

2.7 วัตถุประสงค์

The goal of this project was three fold. First, it aimed to develop a common framework under which one can readily evaluate existing schemes and protocols for sensor networks. Second, it aimed to use this framework to further our understanding about how to design and evaluate schemes and protocols for sensor network applications. Third, with this understanding, we aimed to design *new* simple, efficient, and secure schemes or protocols for sensor network applications.

2.8 ระเบียบวิธีวิจัย

We began by identifying sensor network application areas that are interesting, useful, and in need of cryptographic solutions to ensure secure operations of the sensor network. Then, we focus on schemes and protocols that can be applied to these sensor network applications. The approach has proved fruitful as evidenced by our publication records. These publications explore in depth three constructs, namely blind message authentication codes, CAPTCHAs (Completely Automated Public Turing tests to tell Computers and Humans Apart), and aggregate signatures.

2.9 ผลการวิจัย

As mentioned in the previous section, rather than focusing on the framework issues, we decided instead to focus on schemes and protocols that can be applied to sensor network applications. The constructs we explored in depth were blind message authentication codes, CAPTCHAs, and aggreggate signatures. Each of the constructs results in an international publication. Please see Section 4 for a complete list.

We briefly summarize our research results below. Details can be found in the next section and the publications included in the appendices of this report.

Blind MACs. Blind signatures allow a signer to digitally sign a document without being able to glean any information about the document. We investigate the symmetric analog of blind signatures, namely *blind message authentication codes* (blind MACs). One may hope to get the same efficiency gain from blind MAC constructions as is usually obtained when moving from asymmetric to symmetric cryptosystems. Our main result is a negative one however: we show that the natural symmetric analogs of the unforgeability and blindness requirements cannot be simultaneously satisfied. Faced with this impossibility, we show that blind MACs do exist (under the one-more RSA assumption in the random oracle model) in a more restrictive setting where users can share common state information.

CAPTCHAs. We propose a new construct, the Text-Graphics Character (TGC) CAPTCHA, for preventing dictionary attacks against password authentication systems allowing remote access via dumb terminals. Password authentication is commonly used for computer access control. But password authentication systems are prone to dictionary attacks, in which attackers repeatedly attempt to gain access using the entries in a list of frequently-used passwords. CAPTCHAs (Completely Automated Public Turing tests to tell Computers and Humans Apart) are currently being used to prevent automated ‘‘bots’’ from registering for email accounts. However, current CAPTCHAs are unsuitable for text-based remote access. TGC CAPTCHAs fill this gap.

Aggregate signatures. Secure use of the BGLS [8] aggregate signature schemes is restricted to the aggregation of distinct messages (for the basic scheme) or per-signer distinct messages (for the enhanced, prepend-public-key version of the scheme). We argue that these restrictions preclude interesting applications, make usage of the schemes error-prone and are generally undesirable in practice. Via a new analysis and proof, we show how the restrictions can be lifted, yielding the first truly unrestricted aggregate signature scheme. Via another new analysis and proof, we show that the distinct signer restriction on the sequential aggregate signature schemes of [16] can also be dropped, yielding an unrestricted sequential aggregate signature scheme.

3. เนื้อหางานวิจัย

To avoid duplications, all proofs have been omitted in this section. They can be found in the appendices.

Blind Message Authentication Codes. We define the syntax and security of blind MAC schemes in analogy to those of blind signatures.

Definition 3.1 [**Syntax of a blind MAC scheme.**] A blind MAC scheme $\mathcal{BMAC}$ is a tuple of four polynomial-time algorithms $(\mathsf{Kg}, \mathsf{User}, \mathsf{Tag}, \mathsf{Vf})$ where

– the randomized *key generation* algorithm Kg, on input 1^k with $k \in \mathbf{N}$, outputs a key K.

– User and Tag are possibly randomized interactive algorithms called the *user* and *tagging* algorithm, respectively. The user runs the User algorithm on an initial state containing the security parameter 1^k and a message $\mathrm{MSG} \in \{0,1\}^*$, and lets it interact with the Tag algorithm that is run by the tagger on initial state the key K. [1] At the end of the protocol, the User algorithm either enters the `halt` state and outputs a MAC value τ as its outgoing message, or enters the `fail` state to indicate failure. The Tag algorithm simply enters the `halt` state at the end of the protocol, without generating any output.

– the deterministic *verification* algorithm Vf takes a key K, a message $\mathrm{MSG} \in \{0,1\}^*$ and a MAC value τ as input, and outputs `acc` or `rej` to indicate acceptance or rejection of the MAC value, respectively.

Correctness of a blind MAC scheme requires that for all $k \in \mathbf{N}$ and for all $M \in \{0,1\}^*$, with probability 1 it holds that $\mathrm{St}_{\mathsf{User}} = \texttt{halt}$ and $\mathsf{Vf}(K, \mathrm{MSG}, \tau) = \texttt{acc}$ whenever $K \xleftarrow{\$} \mathsf{Kg}(1^k)$ and $(\mathrm{MSG}_{\mathsf{Tag}}, \mathrm{St}_{\mathsf{Tag}}, \tau, \mathrm{St}_{\mathsf{User}}) \xleftarrow{\$} [\mathsf{Tag}(K) \leftrightarrow \mathsf{User}((1^k, \mathrm{MSG})]$.

Security of blind MACs. Analogously to blind signatures, the security of a blind MAC scheme consists of an unforgeability and a blindness requirement. For unforgeability, the experiment generates a fresh key $K \xleftarrow{\$} \mathsf{Kg}(1^k)$, and runs the adversary A on input 1^k. The adversary can interact in sequential sessions with a tagging oracle that runs the Tag algorithm initialized with key K. At the end of its execution, A outputs m message-tag pairs and wins the game if all messages are different, all tags are valid under key K, and $m > n$, where n is the number of completed tagging sessions during the attack.

Definition 3.2 [**Unforgeability of a blind MAC scheme.**] Let $\mathcal{BMAC} = (\mathsf{Kg}, \mathsf{User}, \mathsf{Tag}, \mathsf{Vf})$ be a blind message authentication scheme. Let $k \in \mathbf{N}$, and let A be a forger with access to the tagging oracle. Consider the following experiment.

Experiment $\mathbf{Exp}^{\text{omu-sa}}_{\mathcal{BMAC},\mathsf{A}}(k)$:
$K \xleftarrow{\$} \mathsf{Kg}(1^k)$; $n \leftarrow 0$
$\{(M_1, \tau_1), \ldots, (M_m, \tau_m)\} \xleftarrow{\$} \mathsf{A}(1^k : \textsc{Tag}(\cdot))$
If $\mathsf{Vf}(K, M_i, \tau_i) = \texttt{acc}$ for all $1 \le i \le m$
and $m > n$ and $M_i \neq M_j$ for all $1 \le i < j \le m$
then return 1 else return 0,

where A's queries to the tagging oracle are answered as follows:

[1] We need to pass 1^k as a parameter to the User algorithm, because otherwise it would no longer be a polynomial-time algorithm if the message is of logarithmic length. Moreover, since the user does not know the key itself, it is reasonable to give it 1^k so that at least it can check whether the tagger is using a key of the correct size.

Oracle $\textsc{Tag}(\mathrm{MSG}_{\mathrm{in}})$:
If $\mathrm{MSG}_{\mathrm{in}} = \perp$ then $\mathrm{St}_{\mathsf{Tag}} \leftarrow K$; $\mathrm{MSG}_{\mathrm{out}} \leftarrow \perp$
else $(\mathrm{MSG}_{\mathrm{out}}, \mathrm{St}_{\mathsf{Tag}}) \xleftarrow{\$} \mathsf{Tag}(\mathrm{MSG}_{\mathrm{in}}, \mathrm{St}_{\mathsf{Tag}}[s])$
If $\mathrm{St}_{\mathsf{Tag}} = \texttt{halt}$ then $n \leftarrow n + 1$
Return $\mathrm{MSG}_{\mathrm{out}}$

The omu-sa advantage of A in breaking $\mathcal{BMAC}$ is defined as the probability that the above experiment returns 1:

$$\mathbf{Adv}^{\text{omu-sa}}_{\mathcal{BMAC},\mathsf{A}}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{omu-sa}}_{\mathcal{BMAC},\mathsf{A}}(k) = 1\,\right] ,$$

and $\mathcal{BMAC}$ is said to be *one-more unforgeable under sequential attacks* (omu-sa-secure) if the advantage $\mathbf{Adv}^{\text{omu-sa}}_{\mathcal{BMAC},\mathsf{A}}(k)$ is a negligible function in the security parameter k for all adversaries A with time complexity polynomial in k. ▌

In the blindness game, the experiment chooses a random bit b and generates a fresh key $K \xleftarrow{\$} \mathsf{Kg}(1^k)$. On input $(1^k, K)$, the adversary A first outputs two messages $\mathrm{MSG}_0, \mathrm{MSG}_1$. The adversary then sequentially interacts with two User sessions, playing the role of the tagger. If $b = 0$, then the first user session is initialized with message MSG_0, and the second with MSG_1; if $b = 1$, then the first session is initialized with message MSG_1, and the second with MSG_0. If both User algorithms accept, the adversary gets to see both resulting tags τ_0, τ_1 for messages $\mathrm{MSG}_0, \mathrm{MSG}_1$. It has to guess the value of b.

We stress that the experiment does not enforce the resulting tags to be valid under key K. While we could include such restriction in the formal security notion, it would be out of touch with reality: the secret key K is not known to the users, so there is nobody to enforce this restriction in the real world. In fact, as we will see in the next section, it is exactly this lack of verifiability of tags that plays a central role in the proof of impossibility of blind MACs. We give a formal blindness definition below.

Definition 3.3 [**Blindness of a blind MAC scheme.**] Let $\mathcal{BMAC} = (\mathsf{Kg}, \mathsf{User}, \mathsf{Tag}, \mathsf{Vf})$ be a blind message authentication scheme. Let $k \in \mathbf{N}$, and let A be an adversary. Consider the following experiment.

Experiment $\mathbf{Exp}^{\text{blind-sa}}_{\mathcal{BMAC},\mathsf{A}}(k)$:
$b \xleftarrow{\$} \{0,1\}$; $K \xleftarrow{\$} \mathsf{Kg}(1^k)$; $((\mathrm{MSG}_0, \mathrm{MSG}_1), \mathrm{St}_{\mathsf{A}}) \xleftarrow{\$} \mathsf{A}(\varepsilon, (1^k, K))$
$(\mathrm{MSG}_{\mathsf{A}}, \mathrm{St}_{\mathsf{A}}, \tau_b, \mathrm{St}_b) \xleftarrow{\$} [\mathsf{A}(\mathrm{St}_{\mathsf{A}}) \leftrightarrow \mathsf{User}((1^k, M_b))]$
$(\mathrm{MSG}_{\mathsf{A}}, \mathrm{St}_{\mathsf{A}}, \tau_{1-b}, \mathrm{St}_{1-b}) \xleftarrow{\$} [\mathsf{A}(\mathrm{St}_{\mathsf{A}}) \leftrightarrow \mathsf{User}((1^k, M_{1-b}))]$
If $\mathrm{St}_0 = \texttt{fail}$ or $\mathrm{St}_1 = \texttt{fail}$ then $\tau \leftarrow \texttt{fail}$
Else $\tau \leftarrow (\tau_0, \tau_1)$
$d \xleftarrow{\$} \mathsf{A}(\tau, \mathrm{St}_{\mathsf{A}})$
If $b = d$ then return 1 else return 0

The blind-sa advantage of A in breaking $\mathcal{BMAC}$ is

$$\mathbf{Adv}^{\text{blind-sa}}_{\mathcal{BMAC},\mathsf{A}}(k) \;=\; 2 \cdot \Pr\left[\,\mathbf{Exp}^{\text{blind-sa}}_{\mathcal{BMAC},\mathsf{A}}(k) = 1\,\right] - 1$$

and $\mathcal{BMAC}$ is said to be *blind under sequential attacks* (blind-sa-secure) if the advantage $\mathbf{Adv}^{\text{blind-sa}}_{\mathcal{BMAC},\mathsf{A}}(k)$ is a negligible function in the security parameter k for all adversaries A with time complexity polynomial in k. ▌

We show that blind MAC schemes simultaneously satisfying the one-more unforgeability and blindness requirements cannot exist. In particular, we demonstrate a universal blindness adversary A and a universal forger F so that for any candidate scheme, one of them always wins with non-negligible probability.

Theorem 3.4 [**Secure blind MAC schemes do not exist.**] Let $\mathcal{BMAC}$ be a blind MAC scheme. Either $\mathcal{BMAC}$ is one-more forgeable under sequential attacks, or it is not blind under sequential attacks.

A construction based on blind signatures. If we allow users to share state, however, secure blind MACs do exist. The main idea for the construction is to store the public key for the base blind signature scheme in the users' common state information. Then, we use the algorithms of the blind signature scheme in a natural way.

Construction 3.5 [A blind MAC scheme for state-sharing users.] Let $\mathcal{BS} = (\mathsf{Kg}_s, \mathsf{User}_s, \mathsf{Sign}, \mathsf{Vf}_s)$ be a blind signature scheme. We associate to it a blind MAC scheme $\mathcal{BMAC} = (\mathsf{Kg}_m, \mathsf{User}_m, \mathsf{Tag}, \mathsf{Vf}_m)$:

- On input 1^k, the key generation algorithm Kg_m runs $\mathsf{Kg}_s(1^k)$ to obtain a key pair (pk, sk), sets $K \leftarrow (pk, sk)$ and returns K.
- On input K, the tagging algorithm Tag starts the interaction with User_m by parsing K as (pk, sk), sends pk to User_m, runs Sign on initial state sk interacting with User_m to completion. It sets its state to whatever Sign does.
- On inputs an initial state 1^k, a message M, and an initial shared-state CSt, the algorithm User_m first receives pk from Tag. If $CSt = \varepsilon$, then User_m sets $CSt \leftarrow pk$. Otherwise, it sets $pk \leftarrow CSt$ and runs User_s on the initial state (pk, M) interacting with Tag until the interaction completes. It sets its state and output to those of User_s.
- On input a key K, a message M, and a MAC value τ, the algorithm Vf_m parses K as (pk, sk), and returns $\mathsf{Vf}_s(pk, M, \tau)$. ▌

The following theorem states that, if the underlying blind signature scheme is one-more unforgeable and *dishonest-key blind*, then the resulting blind MAC scheme is secure. The proof follows directly from the lemmas below. For brevity, we provide only their proof sketches here.

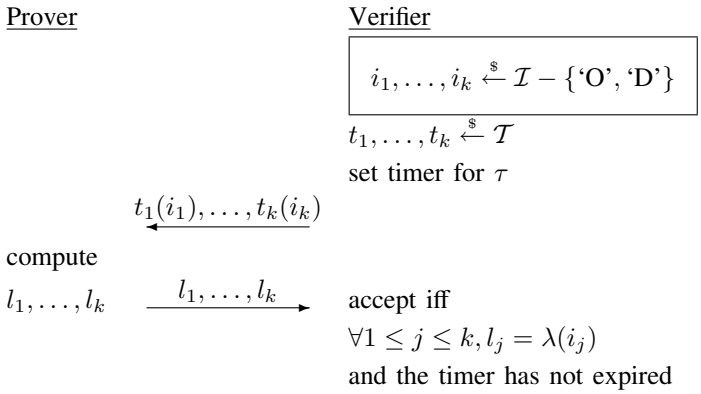
Theorem 3.6 If a blind signature scheme $\mathcal{BS}$ is one-more unforgeable and dishonest-key blind under sequential attacks, then the blind MAC scheme with state-sharing users $\mathcal{BMAC}$ associated to $\mathcal{BS}$ as per Construction 3.5 is one-more unforgeable and blind under sequential attacks.

Text-Graphics Character CAPTCHAs. We describe the TGC CAPTCHAs that we have implemented following the formalization in [1].

Let $\mathcal{I}$ be a set of images of all upper case English characters, $\mathcal{T}$ be a set of transformations on images, λ be the map from an image of a character to the (ASCII ID of) the character portrayed in the image, and τ and k be the security parameters. The TGC CAPTCHA TGC_1 is a tuple $(\mathcal{I}, \mathcal{T}, \lambda, \tau, k)$ defining the test shown in Figure 1. First, the verifier (i.e. server) draws $i_1, \ldots, i_k \xleftarrow{\$} \mathcal{I} - \{\text{'O'}, \text{'D'}\}$ and $t_1, \ldots, t_k \xleftarrow{\$} \mathcal{T}$. Then, it sends to the prover (i.e. user) the transformed images $t_1(i_1), \ldots, t_k(i_k)$ and sets the timer for τ. The prover responds with the labels $l_1, \ldots, l_k$, each of which is (the ASCII ID of) a character in the English alphabet. The verifier accepts if $l_j = \lambda(i_j)$ for all $1 \leq j \leq k$ and if the timer has not expired. It rejects otherwise.

We describe here our choices for TGC_1 for the sets $\mathcal{I}$ and $\mathcal{T}$. The reference image for each character, from the standard X Window System ''9x15'' font, is shown in Figure 2. We use all of the uppercase English characters except 'O' and 'D' which are practically indistinguishable when distorted. The transformation process involves the following steps. First, n_d distracters are chosen uniformly with replacement from the set of all distracter images. In our implementation, we use $n_d = 5$ samples from a set of 26 9x15 bitmaps that share some features with English letters but are easily classified as non-letters by humans.

The TGC CAPTCHA in Random Figlet Fonts $\mathsf{TGC}_2 = (\mathcal{I}, \mathcal{T}, \lambda, \tau, k)$ is similar to TGC_1 shown in Figure 1. The differences between the two are in the choices of the sets $\mathcal{I}$ and $\mathcal{T}$. Specifically, $\mathcal{I}$ is the set of English



รูปที่ 1: Text-Graphics Character CAPTCHAs. An instance of a TGC CAPTCHA is $\mathsf{TGC} = (\mathcal{I}, \mathcal{T}, \lambda, \tau, k)$. The protocol shown is for TGC_1. For TGC_2, the set $\mathcal{I}$ includes both upper and lower case English letters, and the boxed text is replaced by $i_1, \ldots, i_k \xleftarrow{\$} \mathcal{I} - \{\text{'I'}, \text{'L'}, \text{'O'}, \text{'D'}\}$.

รูปที่ 2: Our choice for the reference image set $\mathcal{I}$ for TGC_1. We use only uppercase English characters but omit the characters 'O' and 'D' because they are hard to distinguish when distorted.

---

characters in both upper and lower case excluding 'I', 'L', 'O', and 'D'. The set $\mathcal{T}$ contains the figlet fonts `basic`, `big`, `block`, `broadway`, `colossal`, `cosmic`, `cybermedium`, `doh`, `doom`, `dotmatrix`, `epic`, `fender`, `nancyj`, `ogre`, `pebbles`, `puffy`, `roman`, `rounded`, `starwars`, `stop`, `univers`, and `whimsy` [**?**]. A few examples are shown in Figure 3. Computing $t(i)$ where $t \in \mathcal{T}$ and $i \in \mathcal{I}$ yields character i in the figlet font t.

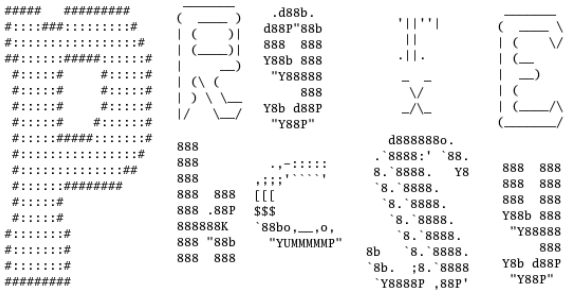
Putting CAPTCHA to Use in SSH. We have implemented a prototype TGC CAPTCHA password authentication method compatible with the SSH user authentication protocol [19]. As a concrete example, we base our implementation on the TGC CAPTCHA TGC_1 and OpenSSH 3.6.1. However, the method could just as easily be based on TGC_2 and/or be incorporated into any SSH-compliant client or server. The latter is so because SSH was specifically designed to allow new user authentication methods to be added in a modular fashion.

Theoretical Results. Intuitively, the following theorem states that each CAPTCHA $\mathsf{TGC} = (\mathcal{I}, \mathcal{T}, \lambda, \tau, k)$ that defined above is secure assuming that the underlying problem $P2_{\mathcal{I},\mathcal{T},\lambda}$ is hard.

Theorem 3.7 Let k be the security parameter, and let $\mathcal{I}, \mathcal{T}, \lambda$ be as previously defined. Let $\delta, \tau, \alpha, \beta$ be non-negative real numbers. Assume that $\mathsf{TGC} = (\mathcal{I}, \mathcal{T}, \lambda, \tau, k)$ is (α, β)-human executable. If $P2_{\mathcal{I},\mathcal{T},\lambda}$ is $(\delta, \tau + O(k))$-hard, then TGC is a (α, β, δ)-CAPTCHA with respect to $P2_{\mathcal{I},\mathcal{T},\lambda}$.

Experimental Results. We performed two experiments to assess the difficulty of our TGC CAPTCHAs for humans and machines.

Experiment 1: Playing against humans. In this experiment, we set out to answer the question of whether our TGC CAPTCHAs are easy enough for humans to be practical complements to password authentication.



รูปที่ 3: Example TGC CAPTCHA characters for TGC_2.

We ran two experiments, 1A and 1B, for TGC_1 and TGC_2, respectively. For each experiment, 20 naive subjects were recruited from the faculty, staff, and students of Thammasat University and the Asian Institute of Technology. Each subject participated in an individual session lasting approximately 5 minutes. They were instructed to maximize their accuracy without regard to time. The instructions were followed by two practice trials with two different sequences of $k = 8$ characters displayed on a $n_c = 80$ by $n_r = 24$ screen. At the end of each trial, the subjects received feedback on whether their response was correct or incorrect. If incorrect, their response and the correct response were displayed.

Following the practice trials were 10 test trials with the same parameters. During the test trials, the subjects' responses and response times were recorded. (They were not told that their response times were being recorded, however.)

For Experiment 1A, we used only upper case English characters excluding 'O' and 'D' and displayed them on noisy screens. For Experiment 1B, we used both upper and lower case English characters excluding 'I', 'L', 'O', and 'D'. In both experiments, the subjects were instructed that they could type either upper case or lower case responses without penalty.

The subjects' average per-character accuracy p_h on the test trials was 0.960 for Experiment 1A and 0.965 for Experiment 1B. Their average word-level accuracy (the number of 8-letter TGC CAPTCHAs answered with 100% accuracy) was 0.765 for Experiment 1A and 0.780 for Experiment 1B. (Assuming independence and $p_h = 0.960$ for 1A and $p_h = 0.965$ for 1B, we would expect a word-level accuracy $(p_h)^k$ of 0.721 for 1A and 0.752 for 1B.)

The fact that naive users achieve such high accuracy rates justifies the use of TGC CAPTCHAs in live systems. Frequent users would very rapidly adapt to the statistics of the character set, achieving even higher accuracy rates.

Experiment 2: Playing against a machine. We ran two experiments, 2A and 2B, for TGC_1 and TGC_2, respectively. In each experiment, we sought to put an upper bound on the difficulty of each TGC CAPTCHA for machines. To this end, we employed an Optical Character Recognition (OCR) system as an adversary against our CAPTCHAs. We selected GOCR [17] because it is open-source, has an active developer community, and runs on a variety of platforms including the UNIX-like operating systems that ship SSH by default.

Using the same parameters as Experiments 1A and 1B, for both Experiment 2A and 2B, we generated 100 TGC CAPTCHAs of length 8, for a total of $2 \times 100 \times 8 = 1600$ text-graphics characters. We then converted each textual display into a bitmap. Each row and column of the bitmap corresponds to a row and column in the text display. We mapped the background text character to white and all other characters to black.

We then built the GOCR 0.39 program from source code using its default configuration, and fed each bitmap directly to the program. In both experiments, we gave GOCR the legal set of characters it should detect, i.e. the 24 characters 'A'–'Z' excluding 'D' and 'O' for Experiment 2A and the 44 characters 'A'–'Z' and 'a'–'z' excluding both upper and lower case versions of 'D', 'I', 'L', and 'O'. We call this the *Naive GOCR* adversary

Scheme	Sign	Aggregate verification process accepts iff
$\mathcal{AS}$-*1* [8]	$\mathrm{H}(m)^x$	$\mathbf{e}(\sigma, g) = \prod_{i=1}^{n} \mathbf{e}(\mathrm{H}(m_i), g^{x_i})$ and $m_1, \ldots, m_n$ all distinct
$\mathcal{AS}$-*2* [8]	$\mathrm{H}(g^x \| m)^x$	$\mathbf{e}(\sigma, g) = \prod_{i=1}^{n} \mathbf{e}(\mathrm{H}(g^{x_i} \| m_i), g^{x_i})$ and $g^{x_1} \| m_1, \ldots, g^{x_n} \| m_n$ all distinct
$\mathcal{AS}$-*3*	$\mathrm{H}(g^x \| m)^x$	$\mathbf{e}(\sigma, g) = \prod_{i=1}^{n} \mathbf{e}(\mathrm{H}(g^{x_i} \| m_i), g^{x_i})$

ตารางที่ 1: The aggregate signature schemes we discuss. Here $\mathbf{e}$: $\mathbf{G}_1 \times \mathbf{G}_2 \to \mathbf{G}_T$ is a bilinear map, g is a generator of $\mathbf{G}_2$ known to all parties, and H: $\{0,1\}^* \to \mathbf{G}_1$ is a hash function. The second column shows the signature of a message m under public key g^x, generated using secret key x. In all cases, a sequence of signatures is aggregated by simply multiplying them in $\mathbf{G}_1$. The third column shows under what conditions the aggregate verification algorithm accepts σ as a valid aggregate signature of messages $m_1, \ldots, m_n$ under public keys $g^{x_1}, \ldots, g^{x_n}$ respectively.

---

to emphasize that different configurations could in principle yield better adversaries. After running Naive GOCR on each image, we classified its response as correct or incorrect.

For Experiment 2A, naive GOCR had a per-character accuracy p_m of 0.278 and 0.314 by the first (conservative) and second (loose) criterion, respectively. The word-level accuracy was 0 by both criteria. For Experiment 2B, without any extraneous noise in the image, naive GOCR had the same per-character accuracy p_m of 0.330 by both the strict and loose evaluation criteria. The word-level accuracy was 0.

Unrestricted Aggregate Signatures. We ask whether there exists a truly unrestricted proven-secure aggregate signature scheme. Namely, there should be no distinctness-based restriction of any kind, whether on messages or enhanced messages. We show that the answer is yes. Our result is a new, direct analysis of the security of enhanced-message signature aggregation which shows that the distinctness condition in the aggregate verification process of $\mathcal{AS}$-*2* —namely that this process rejects if any two enhanced messages are the same— can be dropped without compromising security. In other words, an unrestricted scheme can be obtained by the natural adaptation of $\mathcal{AS}$-*2* in which the distinctness condition in the verification is simply removed and all else is the same. This scheme, which we denote $\mathcal{AS}$-*3*, is summarized in the last row of Table . The fact that $\mathcal{AS}$-*3* is very close to $\mathcal{AS}$-*2* is a plus because it means existing implementations can be easily patched.

We clarify that the security of $\mathcal{AS}$-*3* is not proved in [8]. They prove secure only $\mathcal{AS}$-*1*. The security of $\mathcal{AS}$-*2* is a consequence, but the security of $\mathcal{AS}$-*3* is not. What we do instead is to *directly* analyze security in the case that signatures are on enhanced messages. Our analysis explicitly uses and exploits the presence of the prepended public keys to obtain the stronger conclusion that $\mathcal{AS}$-*3* (not just $\mathcal{AS}$-*2*) is secure.

for practical reasons, $\mathcal{AS}$-*3* is a preferable scheme. But the results of [8] do not prove it secure. Here is an example that helps to see what the problem is. Suppose there was an adversary A that, on input $pk = X$ and without making oracle query m, produced a forgery of the form $(X, m), (X', m'), (X', m'), \sigma$, for some $m' \neq m$ and $X' \neq X$, that was accepted by the verification procedure of $\mathcal{AS}$-*3*. Since the output of A contains repeated enhanced messages, the results of [8] do not allow us to rule out the existence of A. Yet, showing that $\mathcal{AS}$-*3* meets the notion of security that we have defined does require ruling out the existence of such an A.

Theorem 3.8 If the coCDH problem is (t', ϵ')-hard, then the $\mathcal{AS}$-*3* aggregate signature scheme is $(t, q_{\mathrm{S}}, n_{\max}, q_{\mathrm{H}}, \epsilon)$-secure for any $t, q_{\mathrm{S}}, n_{\max}, q_{\mathrm{H}}, \epsilon$ satisfying $\epsilon \geq e(q_{\mathrm{S}} + 1) \cdot \epsilon'$ and $t \leq t' - t_{\exp}(2q_{\mathrm{H}} + 2q_{\mathrm{S}} + 3n_{\max} + 1)$. ▌

Our approach to the proof is different from the one used by [8] to prove that $\mathcal{AS}$-*1* is secure if coCDH is hard. They gave a direct reduction to coCDH, meaning, given an adversary attacking $\mathcal{AS}$-*1* they construct and analyze an adversary attacking coCDH. But, in so doing, they end up duplicating a lot of the proof of the security of the $\mathcal{BLS}$ scheme as given in [10]. Instead, we reduce the security of $\mathcal{AS}$-*3* to the security of $\mathcal{BLS}$. That is, we prove the following:

Lemma 3.9 If the $\mathcal{BLS}$ standard signature scheme is $(t', q'_{\mathrm{S}}, q'_{\mathrm{H}}, \epsilon')$-secure then the $\mathcal{AS}$-*3* aggregate signature scheme is $(t, q_{\mathrm{S}}, n_{\max}, q_{\mathrm{H}}, \epsilon)$-secure for any $t, q_{\mathrm{S}}, n_{\max}, q_{\mathrm{H}}, \epsilon$ satisfying $\epsilon \geq \epsilon'$, $q_{\mathrm{S}} \leq q'_{\mathrm{S}} - n_{\max}$, $q_{\mathrm{H}} \leq q'_{\mathrm{H}}$ and $t \leq t' - t_{\exp} \cdot (q_{\mathrm{H}} + n_{\max} + 1)$. ▌

The theorem follows easily from Lemma 3.9 and known results. Our modular approach yields a simple proof even though we obtain a somewhat stronger result.

หนังสืออ้างอิง

[1] L. von Ahn, M. Blum, N.J. Hopper, and J. Langford, ''CAPTCHA: Using hard AI problems for security,'' Advances in Cryptology – EUROCRYPT 2003, ed. E. Biham, Lecture Notes in Computer Science, vol.2656, pp.294–311, Springer-Verlag, Berlin Germany, May 2003.

[2] M. Bellare, A. Boldyreva, and J. Staddon. Randomness re-use in multi-recipient encryption schemeas. In Y. Desmedt, editor, *PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 85–99. Springer-Verlag, Jan. 2003.

[3] M. Bellare and O. Goldreich. On defining proofs of knowledge. In E. F. Brickell, editor, *Advances in Cryptology – CRYPTO'92*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420. Springer-Verlag, Aug. 1992.

[4] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73. ACM Press, Nov. 1993.

[5] M. Bellare and P. Rogaway. Code-based game-playing proofs and the security of triple encryption. In S. Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*. Springer-Verlag, May 2006. Available as Cryptology ePrint Report 2005/334.

[6] M. Bellare and M. Yung. Certifying permutations: Noninteractive zero-knowledge based on any trapdoor permutation. *Journal of Cryptology*, 9(3):149–166, 1996.

[7] A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Y. Desmedt, editor, *PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer-Verlag, Jan. 2003.

[8] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In E. Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer-Verlag, May 2003.

[9] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. A survey of two signature aggregation techniques. *RSA's CryptoBytes*, 6(2), Summer 2003.

[10] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In C. Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer-Verlag, Dec. 2001.

[11] D. Catalano, D. Pointcheval, and T. Pornin. Trapdoor hard-to-invert group isomorphisms and their application to password-based authentication. *Journal of Cryptology*, 2006. To appear, available from `http://www.di.ens.fr/~pointche/`.

[12] J.-S. Coron. On the exact security of full domain hash. In M. Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer-Verlag, Aug. 2000.

[13] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, Apr. 1988.

[14] R. Hayashi, T. Okamoto, and K. Tanaka. An RSA family of trap-door permutations with a common domain and its applications. In F. Bao, R. Deng, and J. Zhou, editors, *PKC 2004: 7th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2947 of *Lecture Notes in Computer Science*, pages 291–304. Springer-Verlag, Mar. 2004.

[15] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters. Sequential aggregate signatures and multisignatures without random oracles. In S. Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*. Springer-Verlag, May 2006. Available as Cryptology ePrint Report 2006/096.

[16] A. Lysyanskaya, S. Micali, L. Reyzin, and H. Shacham. Sequential aggregate signatures from trapdoor permutations. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 74–90. Springer-Verlag, May 2004.

[17] J. Schulenburg *et al.*, ''GOCR: open-source character recognition, version 0.39,'' 2004. Available at `http://jocr.sourceforge.net/index.html`.

[18] H. Shacham. *New Paradigms in Signature Schemes*. PhD thesis, Stanford University, 2005.

[19] T. Ylonen, ''The secure shell (SSH) authentication protocol.'' IETF RFC 4252, Jan. 2006.

4. ผลที่ได้จากโครงการ

Our results have been published as follows:

- Chanathip Namprempre, Gregory Neven, and Michel Abdalla. A Study of Blind Message Authentication Codes. *Special Section on Cryptography and Information Security in IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E90-A(1):179–186, January 2007.
- Chanathip Namprempre and Matthew Dailey. Mitigating Dictionary Attacks with Text-Graphics Character CAPTCHAs. *Special Section on Cryptography and Information Security in IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E90-A(1):75–82, January 2007.
- Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Unrestricted Aggregate Signatures. In C. Cachin, editor *Automata, Languages and Programming, 34th International Colloquium – ICALP 2007*, volume 4596 of *Lecture Notes in Computer Science*, pages 411–422, Springer-Verlag, July 2007.

5. ภาคผนวก

For completeness, we include in this report, the reprints of all three published papers and a full version of our publication on aggregate signatures. We plan to revise the latter and eventually submit it to Journal of Cryptology when the manuscript is ready.

– Chanathip Namprempre, Gregory Neven, and Michel Abdalla. A Study of Blind Message Authentication Codes. *Special Section on Cryptography and Information Security in IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E90-A(1):179–186, January 2007.

– Chanathip Namprempre and Matthew Dailey. Mitigating Dictionary Attacks with Text-Graphics Character CAPTCHAs. *Special Section on Cryptography and Information Security in IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E90-A(1):75–82, January 2007.

– Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Unrestricted Aggregate Signatures. In C. Cachin, editor *Automata, Languages and Programming, 34th International Colloquium – ICALP 2007*, volume 4596 of *Lecture Notes in Computer Science*, pages 411–422, Springer-Verlag, July 2007.

– Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Unrestricted Aggregate Signatures. IACR Eprint report `http://eprint.iacr.org/2006/285`. Full version of the above publication.

6. บทความสำหรับการเผยแพร่

ในปัจจุบันเครือข่ายอุปกรณ์ส่งสัญญาณได้รับความสนใจเป็นอย่างมาก เนื่องจากหน่วยประมวลผลขนาดเล็กมีราคาต่ำลง จึงมีการติดตั้งหน่วยประมวลผลบนอุปกรณ์ส่งสัญญาณชนิดต่าง ๆ เพื่อให้อุปกรณ์เหล่านี้ทำงานร่วมกันได้ในระบบต่าง ๆ เช่น ระบบควบคุมสัญญาณจราจร ระบบเฝ้าสังเกตสภาวะแวดล้อม ระบบพยากรณ์อากาศ ฯลฯ ลักษณะสำคัญประการหนึ่งของหลายระบบที่มีการนำอุปกรณ์ส่งสัญญาณมาใช้คือ ระบบเหล่านี้จะมีการกระจายอุปกรณ์ส่งสัญญาณไว้ในพื้นที่ที่สนใจศึกษาเพื่อให้อุปกรณ์เหล่านี้เก็บข้อมูลต่าง ๆ เพื่อนำไปทำการวิเคราะห์ต่อไป เป็นที่แน่นอนว่า ในระบบที่มีภารกิจที่สำคัญนั้น ควรมีการยืนยันความน่าเชื่อถือของข้อมูลเหล่านี้ (เช่น ในระบบที่ใช้ในทางการทหาร หรือในระบบวัดสภาวะแวดล้อมในโรงงานปฏิกรณ์ปรมาณู) การปกป้องข้อมูลเหล่านี้ จำเป็นต้องพึ่งพาเทคโนโลยีในทางศาสตร์แห่งการสื่อสารอย่างปลอดภัยเป็นอย่างยิ่ง ความท้าทายที่สำคัญอย่างหนึ่งคือ การนำเทคโนโลยีเหล่านี้มาปรับใช้ในระบบที่มีการใช้อุปกรณ์ส่งสัญญาณที่มีหน่วยประมวลผลขนาดเล็กและมีความสามารถจำกัด ในโครงการนี้ คณะผู้วิจัยได้ทำการออกแบบวิธีการและโปรโตคอลที่สามารถนำมาใช้ในระบบอุปกรณ์ส่งสัญญาณได้ วิธีการและโปรโตคอลที่ออกแบบได้แก่ รหัสยืนยันความแท้จริงของข้อมูลแบบนิรนาม ระบบแคพช่า และระบบลายเซ็นอิเล็กทรอนิกส์แบบผลรวม โดยทั้งสามสิ่งนี้ มีการทำงานที่มีประสิทธิภาพ อีกทั้งทางคณะผู้วิจัยยังได้เขียนข้อพิสูจน์ทางคณิตศาสตร์ เพื่อยืนยันว่าระบบเหล่านี้ให้ความปลอดภัยในการทำงานได้จริงอีกด้วย

There has been much interest in sensor network applications in recent years. With decreasing prices of processors as one of the major driving forces, many applications now equip sensors with processors and let them cooperate to achieve a common goal. Applications of this type include traffic control, eco-system monitoring, and forecasting systems. The most common applications involve scattering sensors in some environment so that they can gather data for further analysis. Clearly, in mission-critical applications, it is undesirable to gather data that are unreliable or potentially-corrupt (maliciously or otherwise). The need to protect information communicated among the sensors can only be addressed by cryptography. The challenge is to adapt cryptographic technology to the special needs of sensor network applications. Under this grant, we have designed schemes and protocols that can be applied to sensor network applications. The schemes and protocols we investigated are blind message authentication codes, CAPTCHAs (Completely Automated Public Turing tests to tell Computers and Humans Apart), and aggregate signatures. The proposed schemes and protocols are not only efficient but also provably secure.