Milagro Multi-Factor Authentication

Masahiro Matsui, Hiroaki Ohtsuka, Tetsutaro Kobayashi, Hironobu Okuyama, Akira Nagai, and Go Yamamoto

Abstract

Apache Milagro (incubating) is an open source project to establish open source software (OSS) for cloud computing. A system designer can choose the M-Pin Authentication Protocol (M-PIN) or the extended M-Pin Authentication Protocol (e-M-PIN) in Milagro Multi-Factor Authentication (Milagro-MFA), which is an authentication system in Apache Milagro (incubating). Additionally, e-M-PIN is a non-interactive protocol and is compatible with password-based Hypertext Transfer Protocol (HTTP) authentication methods such as Basic and Digest Access Authentication since password-based HTTP authentication is also non-interactive. Thus, an authentication system that uses password-based HTTP authentication can be easily migrated to e-M-PIN. We presented e-M-PIN at ApacheCon North America held in May 2016 as a contribution for the OSS community.

Keywords: identity-based authentication, M-PIN, Apache Milagro

1. Introduction

Apache Milagro (incubating) [1] is an open source project focused on new authentication technology and was developed by MIRACL, NTT Innovation Institute, Inc. (NTT i^3), and NTT [2]. One of the authentication systems in Apache Milagro (incubating) is Milagro Multi-Factor Authentication (Milagro-MFA). When using Milagro-MFA, the system designer can choose the M-Pin Authentication Protocol (M-PIN) [3] or the extended M-Pin Authentication Protocol (e-M-PIN). We introduce in this article e-M-PIN, an improved M-PIN authentication protocol that the authors developed.

The common features of M-PIN and e-M-PIN are as follows.

(1) ID-based cryptosystem

M-PIN and e-M-PIN are identity (ID)-based authentication methods that are powered by pairing-based cryptography. A user has his own ID and a secret key, which is related to the ID. The server authenticates the user with the ID, the user secret key, and server secret key.

(2) Multi-Factor Authentication

M-PIN and e-M-PIN are user authentication methods based on the user (client) - web server model. Each user has two secrets; one is a secret key stored in the user's computer, and the other is a PIN (personal identification number) code that the user must remember. Even if one of the secrets is leaked, the security of both protocols is still secure.

(3) No need for specific hardware

The user can login using a web browser. Thus, the server does not need to prepare specific secure hardware such as a hardware token for the user. Each user only needs his own computer or smartphone.

(4) No stored user hashed passwords

The server does not have to store a user's hashed password in order to authenticate the user. Therefore, if the server is compromised, the user's hashed password is not leaked. In contrast, password-based Hypertext Transfer Protocol (HTTP) authentication methods such as Basic and Digest Access Authentication need to store the user's hashed password to authenticate the user. Consequently, leakage of the user's hashed password occurs frequently when the server is compromised.

The difference between M-PIN and e-M-PIN is in the amount of communication.

(5) Interactive or non-interactive protocol

M-PIN is an interactive protocol, and e-M-PIN is a non-interactive protocol. In M-PIN, the server and

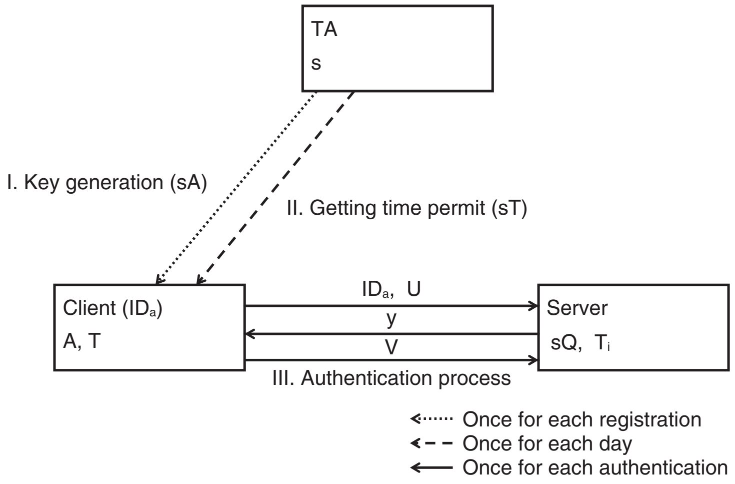

TA
s
I. Key generation (sA)
II. Getting time permit (sT)
Client (ID$_a$)
A, T
ID$_a$, U
y
V
Server
sQ, T$_i$
III. Authentication process
Once for each registration
Once for each day
Once for each authentication

Fig. 1. Entities and processes in M-PIN.

the user have to send data to each other. However, this kind of interactive protocol is not compatible with password-based HTTP authentication, which is non-interactive. Therefore, e-M-PIN is designed as a non-interactive protocol in order to easily migrate from password-based HTTP authentication to e-M-PIN.

To achieve this, we modified M-PIN without reducing its security. We explain the details of M-PIN in section 3 and the points that were improved in section 4.

2. Notations

We define here the notations used in the following sections. The entities are Client, Server, and Trusted Authority (TA). Three functions are discussed: a pairing function (e), which is on an elliptic curve over finite field IF_p, a one-way function (H_q), and a one-way function on a map to a point on the elliptic curve (H_1, H_2). The parameters of these functions are based on the security of cryptography. Therefore, system designers must set the parameters of the functions securely. Recommended parameters can be found in publications of the National Institute of Standards and Technology (NIST) [4], Standards for Efficient Cryptography (SECG) [5], and the Elliptic Curve Cryptography (ECC) Brainpool [6]. Secure parameters have already been implemented in the repository of Apache Milagro (incubating) [7].

- Entities: Client, Server, TA
- q is prime.
- G_1 and G_2 are the q-torsion points on the elliptic curve.
- Z_q is the q-order subgroup of the finite field.
- Pairing e: $G_1 \times G_2 \rightarrow Z_q$
- H_q: $\{0,1\}^* \rightarrow Z_q$
- H_1: $\{0,1\}^* \rightarrow G_1$
- H_2: $\{0,1\}^* \rightarrow G_2$
- P and Q are the respective generators of G_1 and G_2.

3. M-PIN

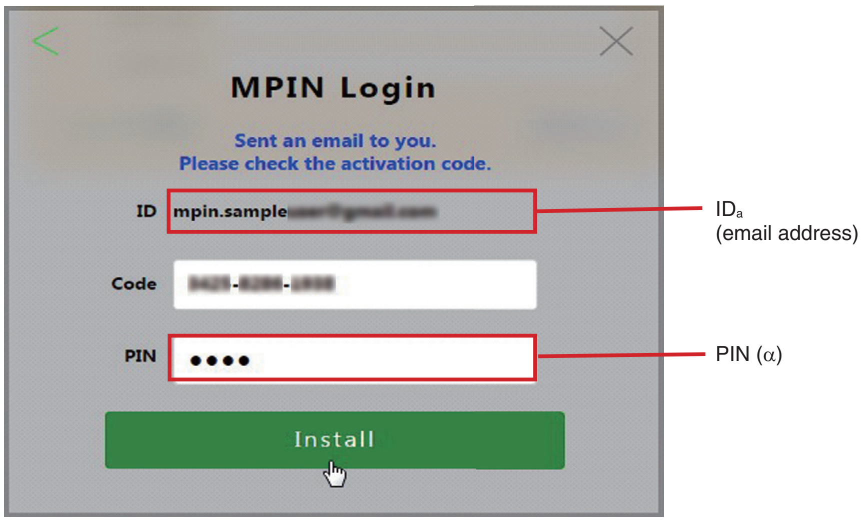
In this section, we provide the details of the M-PIN protocol (**Fig. 1**). M-PIN comprises three steps: Key generation, Getting a time permit, and an Authentication process.

As a preliminary, the server gets a server secret key (sQ) and a today's date (T_i) in advance. Then, the client processes the Key generation and Getting a time permit steps. In the Key generation part, the client gets a client secret key (sA). Key generation is done only once for each user. In Getting a time permit, the client gets today's time permit (sT), which is generated from today's date (T_i). Thus, the Getting a time permit step has to be done once a day for each user as the first step to authenticate the user.

I. Key generation:

1) The client calculates a point (A) on the ellipse curve by using his ID (ID$_a$) and sends it to TA in order to generate a client secret key (sA).
2) The client calculates the token ((s–α)A) using his own PIN (α) and stores the token in local storage.

An image of the M-PIN user registration page is


MPIN Login

Sent an email to you.
Please check the activation code.

ID mpin.sample[redacted]

Code [redacted]

PIN ●●●●

Fig. 2. Image of user registration page of M-PIN.

shown in **Fig. 2**. The user enters his PIN (α), and the browser calculates the token ((s–α)A).

II. Getting a time permit:

1) The first time the client logs in to the server each day, the client sets $T \leftarrow H_1(T_i \| ID_a)$ according to today's date (T_i).
2) The client gets a time permit (sT) by sending T to TA and stores it in local storage.

After completing these steps, the client has his own ID (ID_a) and a client secret key (sA) and a time permit (sT). The client generates a one-time signature (V) with the above information and sends this one-time signature to the server. The server verifies the client with the one-time signature and the server secret key (sQ).

III. Authentication process:

1) The client inputs the PIN (α).
2) The client generates random numbers (x) in Z_q.
3) The client computes $D \leftarrow A+T$ from $A \leftarrow H_1(ID_a)$ and $T \leftarrow H_1(T_i \| ID_a)$.
4) The client computes $U \leftarrow xD$ from D and x.
5) The client sends $\{ID_a, U\}$ to the server.
6) The server generates a random number (y) in Z_q.
7) The server sends y to the client.
8) The client computes $V \leftarrow -(x+y)((s-\alpha)A + \alpha A + sT)$.
9) The client sends $\{V\}$ to the server.
10) The server computes $D \leftarrow H_1(ID_a) + H_1(T_i \| ID_a)$.
11) The server computes $g \leftarrow e(V,Q) \cdot e(U+yD, sQ)$.
12) If g = 1, return true; otherwise, return false.

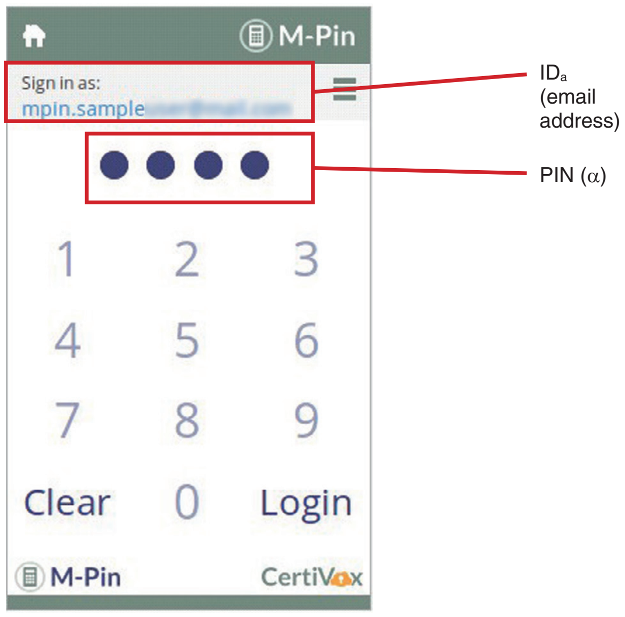

Sign in as:
mpin.sample[redacted]

Fig. 3. Image of M-PIN login page.

An image of the login page of M-PIN and the M-PIN authentication protocol are respectively shown in **Figs. 3** and **4**.

3.1 Advantages of M-PIN

M-PIN has some advantages over password-based HTTP authentication. One advantage is that it uses two-factor authentication. Another is that the server does not have to store a user's hashed password. The

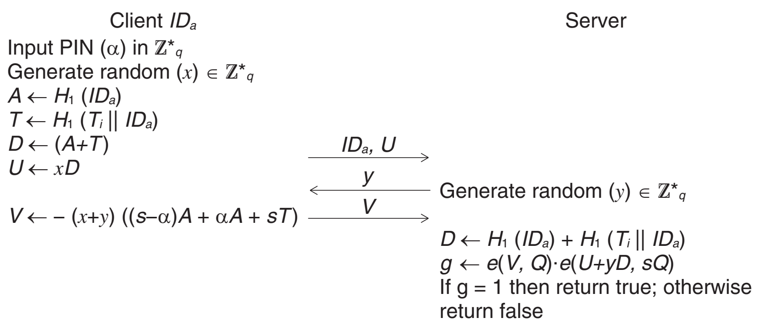

Client ID_a
Server
Input PIN (α) in $\mathbb{Z}^*_q$
Generate random (x) $\in \mathbb{Z}^*_q$
$A \leftarrow H_1\ (ID_a)$
$T \leftarrow H_1\ (T_i \,||\, ID_a)$
$D \leftarrow (A+T)$
$U \leftarrow xD$
ID_a, U
y
Generate random (y) $\in \mathbb{Z}^*_q$
$V \leftarrow -(x+y)\ ((s-\alpha)A + \alpha A + sT)$
V
$D \leftarrow H_1\ (ID_a) + H_1\ (T_i \,||\, ID_a)$
$g \leftarrow e(V, Q)\cdot e(U+yD, sQ)$
If g = 1 then return true; otherwise return false

Fig. 4. M-PIN authentication protocol.

client calculates αA with the user's PIN (α) and adds it to the token ((s–α)A) stored in local storage. With the result of this calculation, the client can create a client secret key (sA). Even if an adversary gets one piece of secret data (PIN (α) or token ((s–α)A)), he cannot calculate the secret key (sA) without having the other piece of secret data. In addition, the server only has to keep the server secret key (sQ) secure. This means that the server does not have to have the user's hashed password. Hence, there is no risk of leakage of the user's hashed password.

3.2 Problem in M-PIN

In M-PIN, the client sends his data (ID_a, U) to the server. Then the server sends a random number (y) to the client, and the client makes a one-time signature (V) that includes y. This means the communication between the client and server requires three passes. However, password-based HTTP authentication, which is one of the most popular forms of authentication, is only one pass. The client sends his password or his hashed password to the server only one time. If the server wants to use both password-based HTTP authentication and M-PIN, the server has to change the communication protocol of the system in order to carry out three-pass communications. This is a major obstacle to be overcome in order to migrate M-PIN.

4. e-M-PIN

In this section, we present the details of the non-interactive authentication protocol e-M-PIN and discuss its advantages and security. This proposed protocol uses one-pass communication between the client and server. This is the same as password-based HTTP authentication. Therefore, the server can use both password-based HTTP authentication and e-M-PIN without any modification of the communication protocol in servers.

However, it is necessary to change the challenge and response method between the client and server, which brings the risk of a replay attack. To maintain the same level of security as M-PIN, e-M-PIN uses some additional information in the authentication process. The proposed protocol can prevent a replay attack by using the current time (CCT: Current Client Time, SCT: Current Server Time) and a nonce (an arbitrary number that can be used only once).

While the Key generation and the Getting time permit processes are the same as that for M-PIN, the preliminary process is slightly different. In addition to getting a private key and a time permit, the server has to set an expiration time (t), which is a time range to accept or not accept a one-time signature. Entities and processes in e-M-PIN are shown in **Fig. 5**.

The authentication process is as below (**Fig. 6**).

III. Authentication process:

1) The client inputs the PIN (α).
2) The client generates random numbers (x) and a nonce in Z_q and gets the current time (CCT) from the client device.
3) The client computes D←A+T from A← H_1 (ID_a) and T← H_1 (T_i||ID_a).
4) The client computes U←xD and W←xA from D, A and x.
5) The client computes y← H_q (ID_a || U || W || nonce || CCT).
6) The client computes V← –(x+y)((s–α)A +αA+sT).
7) The client sends {ID_a,U,W,V,nonce,CCT} to the server.
8) The server gets the current time (SCT).
9) If a nonce exists in the database (DB) that the server has, or the time-gap between SCT and CCT is not in t, the authentication fails.

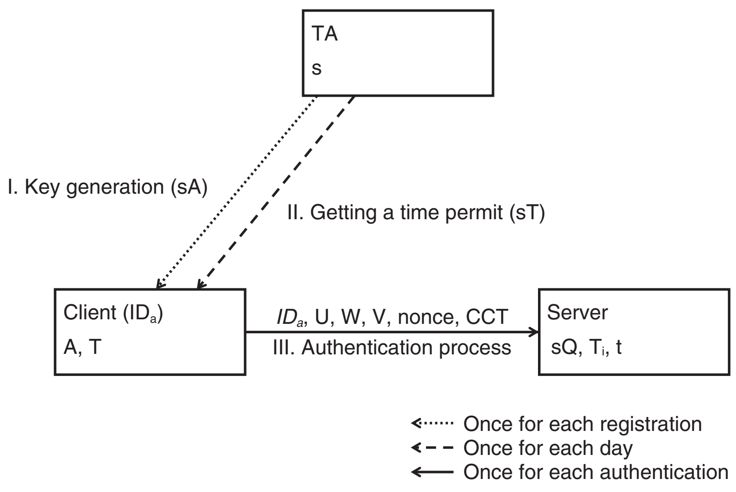
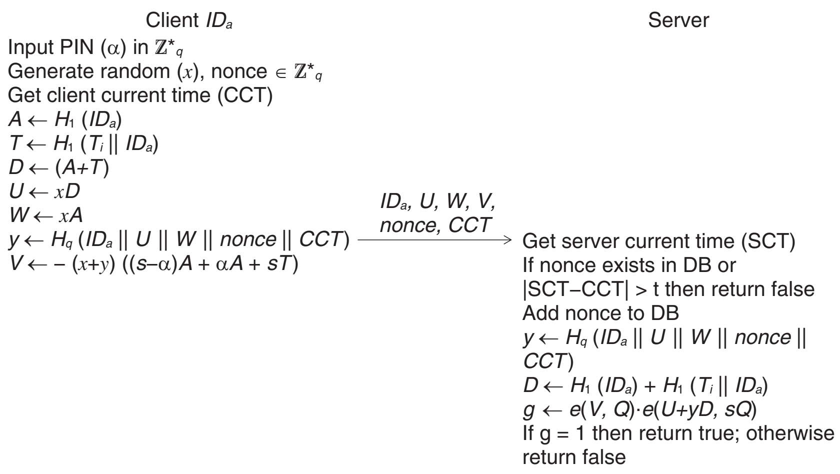

TA
s
I. Key generation (sA)
II. Getting a time permit (sT)
Client (ID_a)
A, T
ID_a, U, W, V, nonce, CCT
III. Authentication process
Server
sQ, T_i, t
Once for each registration
Once for each day
Once for each authentication

Fig. 5. Entities and processes in e-M-PIN.



Client ID_a

Input PIN (α) in $\mathbb{Z}^*_q$
Generate random (x), nonce $\in \mathbb{Z}^*_q$
Get client current time (CCT)
$A \leftarrow H_1(ID_a)$
$T \leftarrow H_1(T_i \,||\, ID_a)$
$D \leftarrow (A+T)$
$U \leftarrow xD$
$W \leftarrow xA$
$y \leftarrow H_q(ID_a \,||\, U \,||\, W \,||\, nonce \,||\, CCT)$
$V \leftarrow -(x+y)\,((s-\alpha)A + \alpha A + sT)$

$\xrightarrow{ID_a,\ U,\ W,\ V,\ nonce,\ CCT}$

Server

Get server current time (SCT)
If nonce exists in DB or
|SCT–CCT| > t then return false
Add nonce to DB
$y \leftarrow H_q(ID_a \,||\, U \,||\, W \,||\, nonce \,||$
$CCT)$
$D \leftarrow H_1(ID_a) + H_1(T_i \,||\, ID_a)$
$g \leftarrow e(V, Q)\cdot e(U+yD, sQ)$
If g = 1 then return true; otherwise
return false

Fig. 6. e-M-PIN authentication protocol.

10) The server adds the nonce to its own DB.
11) The server computes $y \leftarrow H_q(ID_a \,||\, U \,||\, W \,||$ nonce $||$ CCT).
12) The server computes $D \leftarrow H_1(ID_a) + H_1(T_i||ID_a)$.
13) The server computes $g \leftarrow e(V,Q)\cdot e(U+yD,sQ)$.
14) If g = 1, return true; otherwise, return false.

In the challenge and response method, the server first sends a challenge, which is a random number. After receiving the challenge from the server, the client issues a signature with the challenge. Then, the server verifies the signature. If the signature is issued with the server's challenge, the verification is accepted.

In the proposed method (e-M-PIN), the client sends a signature first, without receiving the server's challenge. If the client issues the signature without the challenge, an adversary can initiate a replay attack because the signature becomes a deterministic value. To solve this problem, we use a nonce and the current time.

The nonce works as the server's challenge. The client uses a nonce when he issues a one-time signature (V), and the server checks that the same nonce was never used before. Then, the adversary cannot implement a replay attack.

However, in this protocol, the server has to keep all nonce log data in his DB, which can place a big load

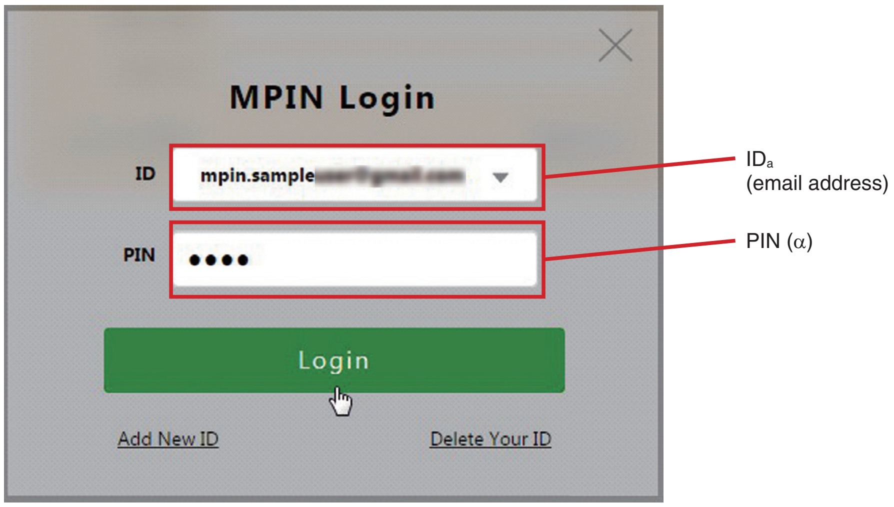


Fig. 7. Login page image of e-M-PIN.

on the server. Therefore, the server sets an expiration time (t) for a one-time signature. The server can then reject an old one-time signature which is |CCT-SCT|<t. With the setting of an expiration time, the server does not have to keep large nonce data for longer than the expiration time. These steps of using a nonce and the current time mean that e-M-PIN can achieve the same level of safety as M-PIN.

The client login page in the authentication process is shown in **Fig. 7**. An ID and PIN are input on the login page. This design is similar to that of the password login page.

4.1 Advantages of e-M-PIN

The advantages of e-M-PIN over M-PIN are as follows; some points have been improved, and there is no reduction in security.

e-M-PIN has almost the same level of security as the previous one. As same as M-PIN, the user splits his secret key into PINα and Token (s–α)A. So, the adversary cannot attack with one factor. Also, the server does not have to store a user's hashed password in the server's storage.

In the e-M-PIN, the communication between client and server is only one time. The client just sends his one-time signature (V) which includes nonce and is made within expiration time (t). So, e-M-PIN can migrate easily from an existing one-pass authentication protocol such as password-based HTTP authentication.

5. Conclusion

We have developed e-M-PIN, an authentication protocol that improves upon M-PIN. e-M-PIN is a non-interactive protocol, which means it is compatible with the traditional password-based HTTP authentication protocol, while the security is equivalent to that of M-PIN. Thus, e-M-PIN is easier to migrate to than M-PIN. e-M-PIN has already been published as open source software in the Apache Milagro (incubating) repository. We presented e-M-PIN at ApacheCon North America held in May 2016 in Vancouver [8]. The demonstration of e-M-PIN showing the ease of migration to it received a positive reaction from visitors. We will continue working to improve Milagro-MFA in order to expand its use in the future.

References

[1] Apache Milagro, Milagro - The Apache Software Foundation!, http://milagro.apache.org/
[2] Press release issued by NTT on May 11, 2016. http://www.ntt.co.jp/news2016/1605e/160511b.html
[3] M. Scott, "M-PIN: A Multi-Factor Zero Knowledge Authentication Protocol," https://cdn2.hubspot.net/hub/230906/file-63954152-pdf/downloads/certivox_labs_mpin.pdf
[4] NIST, "Recommended Elliptic Curves for Federal Government Use," 1999. http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf
[5] SECG, "SEC 2: Recommended Elliptic Curve Domain Parameters," 2010. http://www.secg.org/sec2-v2.pdf
[6] ECC Brainpool (RFC 5639), "ECC Brainpool Standard Curves and Curve Generation," 2005. http://www.ecc-brainpool.org/download/Domain-parameters.pdf

[7] The Apache Software Foundation, Apache Milagro (incubating) Repositories, https://github.com/apache?query=MILAGRO

[8] ApacheCon North America, http://events.linuxfoundation.org/events/apachecon-north-america

Masahiro Matsui

Engineer, Data Security Project, NTT Secure Platform Laboratories.

He received a B.A. in human sciences and an M.S. in global information and telecommunication studies from Waseda University, Tokyo, in 2013 and 2015. He is presently researching information security.

Hironobu Okuyama

Senior Research Engineer, Data Security Project, NTT Secure Platform Laboratories.

He received a B.S. from Tohoku University, Miyagi, in 1990 and an M.S. from Chiba University in 1992. He is presently engaged in research on information security.

Hiroaki Ohtsuka

Research Engineer, Data Security Project, NTT Secure Platform Laboratories.

He received a B.Eng. and M.Eng. from Hokkaido University in 1994 and 1996. He is presently engaged in research on information security.

Akira Nagai

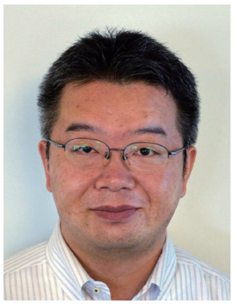
Researcher, Data Security Project, NTT Secure Platform Laboratories.

He received a B.Eng. and M.Eng. from Tokyo University of Science in 2006 and 2008. He is presently researching information security.

Tetsutaro Kobayashi

Senior Research Engineer, Data Security Project, NTT Secure Platform Laboratories.

He received a B.Eng. and M.Eng. from Tokyo Institute of Technology in 1993 and 1995 and a Ph.D. from the University of Tokyo in 2005. He is currently conducting research on information security. He was awarded the SCIS (Symposium on Cryptography and Information Security) 2000 paper prize.



Go Yamamoto

Technology Lead and Associate Director, NTT Innovation Institute, Inc.

He received a B.S., M.S., and Ph.D. in mathematical sciences from the University of Tokyo in 1996, 1998, and 2001. He is currently researching and developing products involving cryptography and distributed systems. He was awarded the SCIS 2007 paper prize.