Identification Systems
Lecture 2

Piotr Syga

2021-03-09

Idea

In order to establish the connection between the passport and the gateway one must:

- confirm the identity of the Reader (privacy)
- confirm the identity of the Chip (access)

. . . yet to establish a secure channel allowing the confirmations we need a shared key.

PACE – idea

J. Bender, M. Fischlin, D. Kügler, *Security Analysis of the PACE Key-Agreement Protocol*

Password Authenticated Connection Establishment

Let us assume that a chip (A) and a reader (B) share a password π chosen at random from a set of cardinality N. We want to generate the secret key K_π as an ephemeral key allowing message exchange using a *controlled* channel in order to establish permanent (session) keys.

Abdalla security model

M. Abdalla, P.-A. Fouque, D. Pointcheval, *Password-Based Authenticated Key Exchange in the Three-Party Setting* (Real or random)

- We assume a set of honest participants $\mathcal{U}$.
- Each participant may run several instances of the key agreement protocol (we denote it $(U; j)$).
- Each pair shares a multi-use password π to generate sessions keys.
- Protocol P is executed between two instances of participants.
- Active adversary model – goal: distinguishing between properly obtained from P keys and random ones.

Protocol execution

Succesful termination – instance $(U; i)$ outputs a key K', the session ID `sid` (fraction of the transcript in PACE), and interlocutor's ID `pid` (empty in PACE for anonymity reasons)

Partners – instances $(A; i)$ and $(B; j)$ if both instances have terminated in accepting state with the same output (any honest, untampered execution should be partnered)

Freshness – an instance $(U; i)$ is *fresh* if there has been no `Reveal(U; i)` query at any point, nor `Reveal(B; j)` for partnering instance and U and B remain uncorrupted.

FS-Freshness – a fresh instance $(U; i)$ that for each `Corrupt(U)` has been no subsequent `Test(U; *)` or no `Send(U; i;m)` at any point.

The Oracles

We assume that the adversary ($\mathcal{A}$) is in full control of the network (may prevent proper communication) and can query the following oracles:

Execute(A; i;B; j) – the users A and B run the protocol for (fresh) instances i and j, producing the transcript of the execution. (*Eavesdropping*)

Send(U; i;m) – starts (U; i) as of receiving message m, producing output (from U's POV) depending on the state of i. (*Active attack*)

Reveal(U; i) – if the instance was terminated in the accepting state, returns the session key of i. (*Key leakage*)

The Oracles

We assume that the adversary ($\mathcal{A}$) is in full control of the network (may prevent proper communication) and can query the following oracles:

Corrupt(U) – the adversary obtains U's π (weak corruption) or U's π and all states of $(U; *)$. (*Node capture*)

Test(U; i) – initializes the oracle with a random bit b. Assume that $\mathcal{A}$ queries about $(U; i)$ during the attack and i ends in accepting state, holding a secret key K. The oracle returns K if b $= 0$ or a random key K' if b $= 1$. (*Adversarial distinction*)

AKE security

The Authenticated Key Exchange security is measured as an advantage of the adversary to 'guess' the value of b for a fresh or fs-fresh instance $(U; i)$.

The Advantage of $\mathcal{A}$

Let t be the number of $\mathcal{A}$ steps (standard steps are also counted), q_e be the upper bound for number if initiated executions, q_h and q_c the number of queries to the external hash oracle and external cipher oracle, respectively. Let $Q = (q_e, q_h, q_c)$ be the set of the upper bounds. We define the AKE advantage of (t, Q)-bounded $\mathcal{A}$ for P as:

$$\mathrm{Adv}_P^{\mathrm{ake}}(\mathcal{A}) = 2\Pr[b == b'] - 1 \ ,$$

$$\mathrm{Adv}_P^{\mathrm{ake}}(t, Q) = \max\left\{\mathrm{Adv}_P^{\mathrm{ake}}(\mathcal{A}) | \mathcal{A} \text{ is } (t, Q)\text{-bounded}\right\} \ .$$

In a similar manner we define ake-fs advantage.

PACE Protocol

A		B
password π		password π
authenticated EC parameters $\mathcal{G} = (a, b, p, q, G, k)$		
$K_\pi = \mathcal{H}(\pi \Vert 0)$ choose $s \leftarrow \mathbb{Z}_q$ $z = \mathcal{C}(K_\pi, s)$		$K_\pi = \mathcal{H}(\pi \Vert 0)$
	$\xrightarrow{\mathcal{G}, z}$	abort if $\mathcal{G}$ incorrect $s = \mathcal{C}^{-1}(K_\pi, z)$
..... Map2Point(s)		
	$\Longleftrightarrow$	
	jointly generate $\widehat{G}$	
choose $y_A \leftarrow \mathbb{Z}_q^*$ $Y_A = y_A \cdot \widehat{G}$		choose $y_B \leftarrow \mathbb{Z}_q^*$ $Y_B = y_B \cdot \widehat{G}$
	$\xleftarrow{Y_B}$	
abort if $Y_B \notin \langle G \rangle \setminus \{0\}$		
	$\xrightarrow{Y_A}$	abort if $Y_A \notin \langle G \rangle \setminus \{0\}$
$K = y_A \cdot Y_B$ $K_{\text{enc}} = \mathcal{H}(K \Vert 1)$ $K_{\text{mac}} = \mathcal{H}(K \Vert 2)$ $K'_{\text{mac}} = \mathcal{H}(K \Vert 3)$ $T_A \leftarrow \mathcal{M}(K'_{\text{mac}}, (Y_B, \mathcal{G}))$		$K = y_B \cdot Y_A$ $K_{\text{enc}} = \mathcal{H}(K \Vert 1)$ $K_{\text{mac}} = \mathcal{H}(K \Vert 2)$ $K'_{\text{mac}} = \mathcal{H}(K \Vert 3)$ $T_B \leftarrow \mathcal{M}(K'_{\text{mac}}, (Y_A, \mathcal{G}))$
	$\xleftarrow{T_B}$	
abort if T_B invalid	$\xrightarrow{T_A}$	abort if T_A invalid
key$= (K_{\text{enc}}, K_{\text{mac}})$ sid $= (Y_A, Y_B, \mathcal{G})$ pid $= \epsilon$		key$= (K_{\text{enc}}, K_{\text{mac}})$ sid $= (Y_A, Y_B, \mathcal{G})$ pid $= \epsilon$

Map2Point

Several versions of Map2Point procedure has been proposed, to obtain a group generator $\hat{G}$. **Diffie-Hellman Mapping**
Both parties calculate $H = x_a x_b G$ with DH exchange of $X_a = x_a G$ and $X_b = x_b G$ and derrive $\hat{G} = sG + H$ with checking for the trivial cases.

DH2Point(s)

A		B
choose $x_A \leftarrow \mathbb{Z}_q^*$		choose $x_B \leftarrow \mathbb{Z}_q^*$
$X_A = x_A \cdot G$		$X_B = x_B \cdot G$
	$\xleftarrow{X_B}$	
	$\xrightarrow{X_A}$	
abort if $X_B = 0$		abort if $X_A = 0$
$H = x_A \cdot X_B$		$H = x_B \cdot X_A$
$\widehat{G} = sG + H$		$\widehat{G} = sG + H$

Map2Point

Several versions of Map2Point procedure has been proposed, to obtain a group generator $\hat{G}$.

Coin-flipping Mapping

A generates $X_a = x_a G$ and presents $H(X_a)$, B transmits similarly obtained X_b (in plain), and A transmits their value, which is checked against the commitment. $H = X_a + X_b$ and $\hat{G} = sG + H$.

Coin2Point(s)

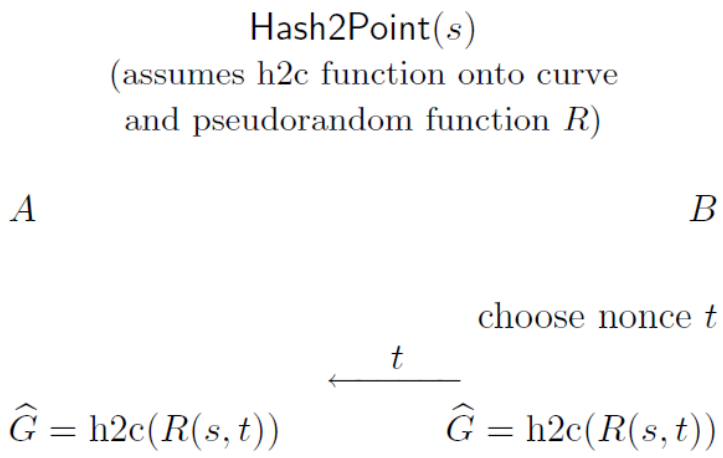
A		B
choose $x_A \leftarrow \mathbb{Z}_q^*$		choose $x_B \leftarrow \mathbb{Z}_q^*$
$X_A = x_A \cdot G$		$X_B = x_B \cdot G$
$C_A = \mathcal{H}(X_A)$	$\xrightarrow{C_A}$	
	$\xleftarrow{X_B}$	
	$\xrightarrow{X_A}$	
		abort if $C_A \neq \mathcal{H}(X_A)$
$H = X_A + X_B$		$H = X_A + X_B$
$\widehat{G} = sG + H$		$\widehat{G} = sG + H$

Map2Point

Several versions of Map2Point procedure has been proposed, to obtain a group generator $\hat{G}$.

Hash-into-the-Curve Mapping

We assume efficient function *h2c* that allows projecting a string directly into the curve and PRNG R. B generates and sends another nonce t and generate a random string using $R(s, t)$.



Map2Point

Several versions of Map2Point procedure has been proposed, to obtain a group generator $\hat{G}$.

Power-to-Group Mapping

A and B map s to element $s^w \mod p$, with $p = wq + 1$ and w being β-smooth for large β. H and $\hat{G}$ may be obtained using interactive generation or reencryption.

Power2Point(s) (with re-encryption)
(for $\mathbb{Z}_p^*$, $p = wq + 1$)

A		B
		choose key K'
		$s' = \mathcal{C}(K', s)$
		where $s = \mathcal{C}^{-1}(K_\pi, z)$
	$\xleftarrow{K'}$	
$s' = \mathcal{C}(K', s)$		
$\widehat{G} = (s')^w \mod p$		$\widehat{G} = (s')^w \mod p$

Security Analysis

The Assumptions

- Random Oracle Model
- Ideal Cipher Model
- MAC $\mathcal{M}$ unforgeable under adaptive CMA
- DH Problem is (t, ε)-hard
- PACE-DH is (t, N, q_I, ε)-hard
- gPACE-DH is (t, N, q_I, ε)-hard

Security Analysis

Requirements specific to Map2Point

- Collision Resistance
- Hardness of DL

additionally

for DH:	PACE-DH
for Coin:	PACE-DH + protection against leakage of X_a and X_b
for Hash:	generation of element using R has to be pseudorandom even with partially manipulated input, and neither of the sides may know the DL of the element
for Power:	DL of s' remains unknown, ideal cipher model

Security Analysis

AKE-Security

Thm. 5.1

Let Map2Point be canonical and and assume that the password is chosen from a dictionary of size N. In the random oracle model and the ideal cipher model we have $\mathrm{Adv}^{\mathrm{ake}}_{\mathrm{PACE}}(t, Q) \leq \frac{q_e}{N} + q_e \mathrm{Adv}^{\mathrm{gPACE-DH}}_{\mathrm{Map2Point}}(t', N, q_h) + q_e \mathrm{Adv}^{\mathrm{forge}}_{\mathcal{M}}(t', 2q_e) + \frac{2q_e N^2 + 8q_e^2 N + q_c q_e}{\min\{q, \mathrm{Range}(\mathcal{H})\}}$, where $t' = t + O(k(q_e^2 + q_h^2 + q_c^2 + k))$.

Security Analysis

Proof Sketch omitting FS and Corrupt queries.

Game 0 : Original attack on the protocol

Game 1 : We abort whenever there are $\pi \neq \pi'$ that yield the same K_π. As there are $\frac{N^2}{2}$ such pairs in ROM
$\Pr[Game0] \leq \Pr[Game1] + \frac{N^2}{2|\mathrm{Range}(\mathcal{H})|}$.

Game 2 : We abort whenever there are $K_\pi \neq K'_\pi$ such that $\mathcal{C}^{-1}(K_\pi, z) = \mathcal{C}^{-1}(K'_\pi, z)$, $\Pr[Game1] \leq \Pr[Game2] + \frac{q_e N^2}{2q}$

Game 3 : We abort whenever there are $K \neq K'$ of two accepting user instances that yield identical encryption or MAC keys.
$\Pr[Game2] \leq \Pr[Game3] + \frac{6q_e^2}{|\mathrm{Range}(\mathcal{H})|}$

Security Analysis

Proof cont.

Game 4 : We replace ideal cipher $\mathcal{C}$ with lazy-sampling of 2 lists. Let THQ be the event that the adversary makes a so-called target hash query about $K||n$ for some,where K is the key some honest user instance has derived. The bound on THQ is made using separate, gPACE-DH game.
$\Pr[Game3] \leq \Pr[THQ] + \Pr[Game4|\neg THQ] + \frac{q_c q_e}{q}$

Game 5 : We even declare the adversary victorious if it ever submits a query $K||n$ for $n \in \{1, 2, 3\}$ to the hash oracle $\mathcal{H}$ such that a user instance $(U; i)$ has computed this key and sent out the final MAC. This cannot happen with probability of a single trial higher than $\frac{q_e}{N}$ (guessing at which query the adversary chooses the correct trial and run honest Map2Point).
$\Pr[THQ] \leq \frac{q_e}{N} + q_e(\mathrm{Adv}^{\mathrm{gPACE-DH}}_{\mathrm{Map2Point}}(t', N, q_h))$

Security Analysis

Proof cont.

Game 6 : We replace K_{enc} and K_{mac} in queries by random keys. Note that, since we assume that the adversary never makes a hash query about a DH-key of an accepting user instance, this simulation is perfect unless there is an accepting instance $(U; i)$ having the same DH-key as another instance $(U'; j)$ and the instances are not partnered. If they are it breaches canonicality. Since we guess the first Execute query with probability $1/q_e$ and the find a valid forge, we prevent the user from submitting K to MAC Oracle. Hence, $\mathcal{A}$ can only inject values $(Y_A; G)$, $(Y_B; G)$,$(Y'_A; G')$, $(Y'_B; G')$ with valid MACs generated by honest users in other executions, but since we guessed and K is determined, y_* at random.
$\Pr[Game5|\neg THQ] \leq \Pr[Game6] + \frac{2q_e}{q} + q_e \mathrm{Adv}^{\mathrm{forge}}_{\mathcal{M}}(t', 2q_e)$

Final Game : Al keys returned in Test are random, hence the advantage is bounded by $\frac{1}{2}$.

Final notes

- Forward security requires modification of gPACE-DH problem
- PACE uphold untraceability in a restricted sense – issuers metadata like signature of authentication signautre or EC used allows partial tracking
- Unlinkability between two eavesdropped communication is provided
- Non-transferability follows from the fact that the terminal could generate the data in a communication itself with the help of the password

Further reading

- V. Shoup: Lower Bounds for Discrete Logarithms and Related Problems,
- V. Shoup: Sequences of Games: A Tool for Taming Complexity in Security Proofs,
- Federal Office for Information Security (BSI): Advanced Security Mechanism for Machine Readable Travel Documents – Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI),
- M. Abdalla, P.-A. Fouque, D. Pointcheval: Password-based authenticated key exchange in the three-party setting,
- Th. Icart: How to Hash Into Elliptic Curves,
- A. Shallue, Ch. van de Woestijne: Construction of Rational Points on Elliptic Curves over Finite Fields.,
- C. Blundo et al.: Resettable and Non-Transferable Chip Authentication for E-Passports,
- L. Hanzlik et al.: Mutual Chip Authentication,
- L. Hanzlik, M. Kutyłowski: Chip Authentication for E-Passports: PACE with Chip Authentication Mapping v2.