Advances in Privacy Aware Authentication

Mirosław Kutyłowski
joint work mostly with Lucjan Hanzlik, Kamil Kluczniak

Wrocław University of Technology, Poland

IFIP SEC 2015, Hamburg
invited talk

ID documents today

E-Passport

Necessity for ID documents with a chip

- traditional security printing is not reliable enough:
  - race between authorities and sophisticated forgers
  - personal ID documents should be used for years
- **cryptographic protection – independent and relatively long lasting**

E-Passport

Identity document with a memory chip - a simplest solution

- the printed data stored also on the chip, organized in so-called *data groups*
- data groups signed by the document issuer

Privacy problems

- personal data signed by the state authorities are attractive for **illegal trading** – quality is guaranteed!
- for durability reasons, the chip of the e-passport should communicate via a wireless interface
  – so **skimming is possible**

Basic Access Control

basic protection against skimming

BAC mechanism

- based on a secret symmetric key K_{Enc} shared by the reader and the e-Passport
- K_{Enc} derived by hashing some basic personal data printed on the chip
- mutual authentication: the reader and the terminal mutually prove that they know K_{Enc}
- the session key derived from random strings chosen by the e-Passport

attacks

- low entropy of K_{Enc} $\Rightarrow$ it can be guessed
  $\Rightarrow$ easy offline attacks on recorded communication
- once the adversary learns K_{Enc}, then he can access all data shown by the e-Passport

Basic Access Control
threats

Consequences

Basic Access Control is not a reliable protection of personal data transmitted over a wireless channel.

It is only making access to personal data less straightforward.

... but better BAC than nothing!

Active Authentication

AA basics

purpose secure against cloning the e-Passports
– the passports with BAC can be easily cloned

mechanism a secret key in the e-Passport, the corresponding public key in a data group
a challenge-and-response protocol for showing possession of the secret key

AA and privacy?

- even more privacy threats!
- a reader may prove against third parties that it has interacted with a given e-Passport

Extended Access Control

idea

Background

- high quality biometric data in the e-Passport increase substantially reliability of identification with identity documents
- . . . **but one can expose sensitive data to malicious processing**
- for standard data this is not a problem: they are printed on the passport and can be read anyway

> if biometric data are to be used in the e-Passport, then they have to be well secured against misuse

Extended Access Control
idea

ICAO

protecting sensitive data: Extended Access Control as an option

EAC components

Chip Authentication: the chip gets authenticated, additionally a shared session key is established
the chip's public key used, DH key exchange, implicit authentication

Terminal Authentication: the terminal and its rights (to read sensitive data) checked
authentication via signing a challenge, signature verification based on a chain of CVC certificates

German personal ID card

Main components

- **Terminal Authentication** - checking terminal's access rights
- **Chip Authentication** - checking originality of a chip
- **Restricted Identification** - anonymous authentication
- **PACE** - enabling chip operation with a password

as well as place for qualified signatures

Specifications:

BSI Technische Richtlinie 03110: Advanced Security Mechanisms for Machine Readable Travel Document

Terminal Authentication v. 2

protocol specification of BSI

	terminal		e-ID chip
		$\xrightarrow{cert(PK_{PCD})}$	
1.			verify $cert(PK_{PCD})$, extract PK_{PCD}
2.	choose $\widetilde{SK_{PCD}}$ at random $\widetilde{PK_{PCD}} := g^{\widetilde{SK_{PCD}}}$ compute commitment $Comp(\widetilde{PK_{PCD}})$		
		$\xrightarrow{Comp(\widetilde{PK_{PCD}})}$	
3.			choose r at random
		$\xleftarrow{r}$	
4.	$s := Sign_{\widetilde{SK_{PCD}}}(ID_{PICC} \mid r \mid Comp(\widetilde{PK_{PCD}}) \mid)$		
		$\xrightarrow{s}$	
5.			verify s

Chip Authentication

	terminal		e-ID chip
			static key pair (SK_{PICC}, PK_{PICC})
6.		$\xleftarrow{PK_{PICC}}$	
7.		$\xrightarrow{\widetilde{PK_{PCD}}}$	
8.	$\mathcal{K} := (PK_{PICC})^{\widetilde{SK_{PCD}}}$		$\mathcal{K} := (\widetilde{PK_{PCD}})^{SK_{PICC}}$
9.			choose r' at random $\mathcal{K}_{MAC} := Hash_3(\mathcal{K}, r')$ $TAG := MAC_{\mathcal{K}_{MAC}}(\widetilde{PK_{PCD}})$
		$\xleftarrow{TAG, r'}$	
10.	$\mathcal{K}' := Hash_1(\mathcal{K}, r')$ $\mathcal{K}_{MAC} := Hash_3(\mathcal{K}, r')$		
11.	$TAG \stackrel{?}{=} MAC_{\mathcal{K}_{MAC}}(\widetilde{PK_{PCD}})$		

Password Authenticated Connection Establishment

1. establishes an authenticated encrypted channel only if the correct password given
2. main purpose is to secure wireless communication
3. password guessing as hard as possible:
   — a reader interacting with a chip may try only one password per session
4. implemented in German personal ID cards
5. decided to be obligatory for biometric passports in the EU
6. developed by German BSI security authority,
   a later version with French modifications

PACE-GM (PACE General Mapping)

e-ID chip		reader
π		π typed in by the owner
$K_\pi := H(0\|\|\pi)$		$K_\pi := H(0\|\|\pi)$
choose $s \leftarrow \mathbb{Z}_q$		
$z := \mathrm{ENC}(K_\pi, s)$		
	$\xrightarrow{\mathcal{G}, z}$	abort if $\mathcal{G}$ incorrect
		$s := \mathrm{DEC}(K_\pi, z)$
choose $y_A \leftarrow \mathbb{Z}_q^*$		choose $y_B \leftarrow \mathbb{Z}_q^*$
$Y_A := g^{y_A}$		$Y_B := g^{y_B}$
	$\xleftarrow{Y_B}$	
abort if $Y_B \notin \langle g \rangle \setminus \{1\}$	$\xrightarrow{Y_A}$	abort if $Y_A \notin \langle g \rangle \setminus \{1\}$
$h := Y_B^{y_A}$, $\hat{g} := h \cdot g^s$		$h := Y_A^{y_B}$, $\hat{g} := h \cdot g^s$
choose $y'_A \leftarrow \mathbb{Z}_q^*$		choose $y'_B \leftarrow \mathbb{Z}_q^*$
$Y'_A := \hat{g}^{y'_A}$		$Y'_B := \hat{g}^{y'_B}$
	$\xleftarrow{Y'_B}$	
check $Y'_B \neq Y_B$	$\xrightarrow{Y'_A}$	check $Y'_A \neq Y_A$
$K := {Y'_B}^{y'_A}$		$K := {Y'_A}^{y'_B}$

PACE-IM (PACE Integrated Mapping

in additive notation

e-ID chip		reader
π - password,		π password typed-in by the owner
choose s at random		
	$\xrightarrow{z}$	
$z := \mathrm{ENC}(\pi, s)$		
		$s := \mathrm{DEC}(\pi, z)$
		choose β at random
	$\xleftarrow{\beta}$	
$\hat{G} = \mathrm{Encoding}(\mathrm{Hash}(s, \beta))$ choose $x \leftarrow \mathbb{Z}_q$ at random $X := x \cdot \hat{G}$		$\hat{G} = \mathrm{Encoding}(\mathrm{Hash}(s, \beta))$
	$\xrightarrow{X}$	
		choose $y \leftarrow \mathbb{Z}_q$ at random $Y := y \cdot \hat{G}$
	$\xleftarrow{Y}$	
$Z = x \cdot Y$		$Z = y \cdot X$
	...	

Integrating PACE with Chip Authentication

ChA-CAM according to ICAO

Card		Reader
π, x_A, $X_A = g^{x_A}$		π
random s chosen	$\xrightarrow{ENC(K_\pi, s)}$	retrieve s
choose $y_A \leftarrow \mathbb{Z}_q^*$		choose $y_B \leftarrow \mathbb{Z}_q^*$ $Y_B := g^{y_B}$
$Y_A := g^{y_A}$		
abort if ... $h := Y_B^{y_A}$, $\hat{g} := h \cdot g^s$ choose $y'_A \leftarrow \mathbb{Z}_q^*$	$\xrightarrow{Y_A}$	abort if ... $h := Y_A^{y_B}$, $\hat{g} := h \cdot g^s$ choose $y'_B \leftarrow \mathbb{Z}_q^*$
$Y'_A := \hat{g}^{y'_A}$	$\xleftarrow{Y'_B}$ $\xrightarrow{Y'_A}$	$Y'_B := \hat{g}^{y'_B}$
$K := {Y'_B}^{y'_A}$ ...	...	$K := {Y'_A}^{y'_B}$ ...
$w := y_A / x_A$	$\xrightarrow{E_{K'_{SC}}(w, cert_A))}$	decrypt with K'_{SC} check certificate $cert_A$ abort if $X_A^w \neq Y_A$

Privacy by Design for eID

Password derivation

a recorded transcript of interaction between the reader and an eID should not be useful for offline dictionary attacks – i.e. trying all possible passwords

Example: PACE-IM:

e-ID chip		reader
π - password,		π password typed-in by the owner
choose s at random		
$z := \mathrm{ENC}(\pi, s)$	$\xrightarrow{z}$	
		$s := \mathrm{DEC}(\pi, z)$ choose β at random
	$\xleftarrow{\beta}$	
$\hat{G} = \mathrm{Encoding}(\mathrm{Hash}(s, \beta))$ choose $x \leftarrow \mathbb{Z}_q$ at random $X := x \cdot \hat{G}$		$\hat{G} = \mathrm{Encoding}(\mathrm{Hash}(s, \beta))$
	$\xrightarrow{X}$	
		choose $y \leftarrow \mathbb{Z}_q$ at random $Y := y \cdot \hat{G}$
	$\xleftarrow{Y}$	
$Z = x \cdot Y$		$Z = y \cdot X$
	...	

Simultability

no transferable proof of interaction

any proof of interaction with an eID (the interaction record plus some private values of the terminal) is unreliable, since the terminal can forge it (simulate)

Example: Chip Authentication

	terminal		e-ID chip
	SK_{PCD} chosen at random		static key pair (SK_{PICC}, PK_{PICC})
6.		$\xleftarrow{PK_{PICC}}$	
7.		$\xrightarrow{\widetilde{PK_{PCD}}}$	
8.	$\mathcal{K} := (PK_{PICC})^{\widetilde{SK_{PCD}}}$		$\mathcal{K} := (\widetilde{PK_{PCD}})^{SK_{PICC}}$
9.			choose r' at random $\mathcal{K}_{MAC} := Hash_3(\mathcal{K}, r')$ $TAG := MAC_{\mathcal{K}_{MAC}}(\widetilde{PK_{PCD}})$
		$\xleftarrow{TAG, r'}$	
10.	$\mathcal{K}' := Hash_1(\mathcal{K}, r')$ $\mathcal{K}_{MAC} := Hash_3(\mathcal{K}, r')$		
11.	$TAG \stackrel{?}{=} MAC_{\mathcal{K}_{MAC}}(\widetilde{PK_{PCD}})$		

Tracing

- simultability alone does not mean that an eID cannot be traced: the eavesdropper may observe that some eID is really executing the protocol
- **for an eavesdropper the real transmission traces should not be linkable with eIDs or their pseudonyms**

Erroneous execution

privacy should not be endangered when a terminal or communication line are attacked

Particular attack scenarios:

- manipulating communication: interruption, reset, injecting or removing messages
- replacing terminals or malicious terminals not executing the protocol properly

Erroneous execution

Example: PACE-GM

e-ID chip		reader
π		π typed in by the owner
$K_\pi := H(0\Vert\pi)$		$K_\pi := H(0\Vert\pi)$
choose $s \leftarrow \mathbb{Z}_q$		
$z := \mathrm{ENC}(K_\pi, s)$	$\xrightarrow{z}$	
		$s := \mathrm{DEC}(K_\pi, z)$
choose $y_A \leftarrow \mathbb{Z}_q^*$		choose $y_B \leftarrow \mathbb{Z}_q^*$
$Y_A := g^{y_A}$		$Y_B := g^{y_B}$
	$\xleftarrow{Y_B}$	
abort if $Y_B \notin \langle g \rangle \setminus \{1\}$	$\xrightarrow{Y_A}$	abort if $Y_A \notin \langle g \rangle \setminus \{1\}$
$h := Y_B^{y_A}$, $\hat{g} := h \cdot g^s$		$h := Y_A^{y_B}$, $\hat{g} := h \cdot g^s$
choose $y'_A \leftarrow \mathbb{Z}_q^*$		choose $y'_B \leftarrow \mathbb{Z}_q^*$
$Y'_A := \hat{g}^{y'_A}$		$Y'_B := \hat{g}^{y'_B}$
$Y'_A := \hat{g}^{y'_A}$	$\xleftarrow{Y'_B}$	$Y'_B := \hat{g}^{y'_B}$
	$\xleftarrow{Y'_B}$	
$K := {Y'_B}^{y'_A}$		$K := {Y'_A}^{y'_B}$

Weak randomness

- **If randomness is weak, then the whole security may by an illusion.**
- A malicious provider can install weak randomness to steal secrets and get access to the user's data.
- An attack may concern the randomness used on the eID **or** on the terminal.

This is a likely threat in large scale systems.

Example protection:

Lucjan Hanzlik, Przemysław Kubiak, Mirosław Kutyłowski: Stand-by Attacks on E-ID Password Authentication. INSCRYPT 2014, LNCS 8957

Restricted Identification

Restricted Identification concept

Domains
each domain is an autonomous system such that

- user's personal data are **processed only within the system** (unless a special event occurs)
- within a domain the user appears under his **domain specific identity/pseudonym**
- it should be **infeasible to link** identities of one user in two different domains

Background

- full disclosure of identity is not really necessary
- unnecessary data flow is a privacy risk
- a kind of privacy-by-design

Origin: Austrian concept of sectors

Idea of sectors/domains

1. each sector is a different public sector/public IT system

   Sector examples

   - health care system
   - citizen-police contacts
   - children protection
   - psychological hotline
   - . . .

2. a “citizen card” can automatically generate a password for each sector
3. a central server can compute the password for each citizen/sector combination
4. the password sent by the user is compared against the password created in the central system

a solution based on symmetric cryptography, replay attacks possible

Restricted Identification:

1. e-ID card computes a unique password for each domain
2. the terminal of the domain:
   a) checks that it is talking with an e-ID card
   b) receives a password
   c) checks the password against its blacklist

Restricted Identification

Core RI procedure
(notation according to BSI specification)

Terminal		e-ID chip
holds $\mathcal{K}'$		holds $\mathcal{K}'$
$\sigma := \mathrm{ENC}_{\mathcal{K}'}(PK_{sector})$	$\xrightarrow{\sigma}$	
		$PK_{sector} := \mathrm{DEC}_{\mathcal{K}'}(\sigma)$
		$I_{ID}^{sector} := \mathrm{Hash}(\,(PK_{sector})^{SK_{ID}})$
		$\sigma' := \mathrm{ENC}_{\mathcal{K}'}(I_{ID}^{sector})$
$I_{ID}^{sector} := \mathrm{DEC}_{\mathcal{K}'}(\sigma')$	$\xleftarrow{\sigma'}$	
check if I_{ID}^{sector} is on sector's black-list		

$\mathcal{K}'$ is a shared key that must be established **before** running RI

Security background

- since the chip is assumed to be secure, we have to believe that the eID really sends $I_{ID}^{sector} := \mathrm{Hash}((PK_{sector})^{SK_{ID}})$ using its private RI key SK_{ID}

Blacklist

- a list of values $\mathrm{Hash}((PK_{sector})^x)$, where x belongs to a banned person

Blacklisting a user

- the Issuing Authority holds the public key $PK = g^x$ of that user
- $PK_{sector} = g^{r \cdot R}$, where
  - r is known to the Issuing Authority
  - R is known to the domain authority
- two steps:
  - the Issuing Authority computes $P_1 = PK^r$
  - the domain authority computes P_1^R

note that $P_1^R = PK^{r \cdot R} = g^{x \cdot r \cdot R} = (g^{r \cdot R})^x = (PK_{sector})^x$

Blacklisting properties:

- the Issuing Authority does not learn the password of the revoked user
- **the terminal has to know that it is really talking with a valid eID**
  otherwise a random response would be accepted as a valid pseudonym – it is unlikely that it appear on the blacklist

Challenge

- the terminal must check that it is talking with a valid eID
- **there are many authentication protocols – but how to hide the identity of the chip?**
  standard solutions use something (e.g. a public key) that would link RI passwords in different domains

Group key

Design decision

- authentication of an eID via Chip Authentication with a **group key**
  it does not mean using group signatures
- a large number of eIDs share the same group key
  – a big *anonymity set*

Quotation
One of the designers said:

> *"... this is an assumption that all chips of eID are tamper-resistant ... "*

Realistic attack assumptions

Are group keys really protected?

- a really powerful adversary can break into an eID chip and read its secrets
  – breaking into just one eID of the group is enough!
- if a group key has to be installed in a large number of devices, it must be stored and protected outside the eIDs
- it suffices to provide just one tampered raw eID for personalization – it would reveal the secret (group key) in response to a secret command

what would be the consequences?

Attack 1: creating a fake ID

A fake eID

- contains a valid group key
- provides a random password during execution of the RI protocol

Properties

- **the fake eID works as long as RI is used**
- **impossible to blacklist the fake eID**

Attack 2: account access

observation by Lucjan Hanzlik

A powerful adversary

- learns the group key
- eavesdrops the communication with a domain server

Observation

- **on the side of the eID, Chip Authentication derives the session key with the group key - no ephemeral random values used**
- **so the Adversary can derive the session key as well!**
- **the Adversary can decrypt the ciphertext and get the domain password of this user**

Attack 2: account access
observation by Lucjan Hanzlik

Attack potential

an attacker may login to the user's account after a purely passive attack

It looks like an obvious trapdoor in the German personal identity cards.

Chip Authentication - Restricted Identification

Goal

- no group key
- **authentication of the chip based on the RI secret key**

ChARI protocol

A protocol published in:

Lucjan Hanzlik, Kamil Kluczniak, Przemysl aw Kubiak, Miroslaw Kutylowski: Restricted Identification without Group Keys. IEEE TrustCom 2012: 1194-1199

Lucjan Hanzlik, Mirosław Kutyłowski: Restricted Identification Secure in the Extended Canetti-Krawczyk Model. J. UCS 21(3): 419-439 (2015)

ChARI protocol
Terminal Authentication

- Terminal Authentication is essentially the same as in the German EAC
- eID chip learns PK_{sector} from the terminal's certificate

ChARI protocol

Chip Authentication + Restricted Identification – part 1

	terminal		eID
6.			choose b at random
			$\widetilde{I_{ID}^{sector}} := (PK_{sector})^{b \cdot SK_{ID}}$
		$\xleftarrow{\widetilde{I_{ID}^{sector}}}$	
7.	$\mathcal{K} := (\widetilde{I_{ID}^{sector}})^{\widetilde{SK_{PCD}}}$ choose r' at random, $\mathcal{K}_{MAC} := Hash_1(\mathcal{K}, r')$ $\mathcal{K}_{\mathrm{ENC}} := Hash_2(\mathcal{K}, r')$		$\mathcal{K} := (\widetilde{PK_{PCD}})^{b \cdot SK_{ID}}$
8.	$TAG := MAC(\mathcal{K}_{MAC}, \widetilde{I_{ID}^{sector}})$		
		$\xrightarrow{TAG, r'}$	
9.			$\mathcal{K}_{MAC} := Hash_1(\mathcal{K}, r')$ $\mathcal{K}_{\mathrm{ENC}} := Hash_2(\mathcal{K}, r')$ $TAG \stackrel{?}{=} MAC(\mathcal{K}_{MAC}, \widetilde{I_{ID}^{sector}})$

ChARI protocol
Chip Authentication + Restricted Identification – part 2

	terminal		eID
10.			$\sigma := \mathrm{ENC}_{\mathcal{K}_{\mathrm{ENC}}}(cert(I_{ID}^{sector}))$ or $\sigma := \mathrm{ENC}_{\mathcal{K}_{\mathrm{ENC}}}(r)$ if white/black-list used $\sigma' := \mathrm{ENC}_{\mathcal{K}_{\mathrm{ENC}}}(b)$
		$\xleftarrow{\sigma,\sigma'}$	
11.	$z := \mathrm{DEC}_{\mathcal{K}_{\mathrm{ENC}}}(\sigma)$ $b := \mathrm{DEC}_{\mathcal{K}_{\mathrm{ENC}}}(\sigma')$ $I_{ID}^{sector} := (\widetilde{I_{ID}^{sector}})^{b^{-1}}$ verify that I_{ID}^{sector} on white/black list or verify z		

- the trick is to randomize the sector identifier
- at the end the eID is obliged to derandomize it

Pairing RI

A new solution:

Lucjan Hanzlik,
Cryptographic Protocols for Modern Identification Documents.

PhD Dissertation, submitted in 2015
in Institute of Computer Science, Polish Academy of Sciences

Setup:

1. $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ – a bilinear map group, generators $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$
2. $z \in \mathbb{Z}_q^*$ chosen at random
3. public keys $Z_1 = g_1^z$, $Z_2 = g_2^z$,
4. secret key: z
   public key: (Z_1, Z_2) (and a proof that they are created as described)

an eID joins the system:

- an interactive protocol between the eID and the Issuer holding z
- result:
  - the eID gets a secret key: sk_1, $sk_2 = g_1^{1/(z+sk_1)}$ (i.e. a kind of Boneh-Boyen signature)
  - Issuer: a *revocation token* enabling revocation of the user
- the Issuer does not learn sk_1, sk_2

domain parameters:

- r chosen at random
- $g_{dom} = g_2^r, \quad Z_{dom} = Z_2^r$
- the public parameters are:
  - g_{dom}, Z_{dom}
  - Issuer's certificate for g_{dom}, Z_{dom}
  - a proof that g_{dom}, Z_{dom} have been created correctly

eID domain specific pseudonym:

- $dnym := e(g_1, Z_{dom})^{sk_1}$

eID authenticates itself:
a non-interactive zero knowledge proof that the eID knows α, β such that:

- $dnym = e(g_1, Z_{dom})^{\alpha}$
- $\beta = g_1^{1/(z+\alpha)}$

Lucjan Hanzlik proposes **a concrete realization** such that

- on the eID chip: a few exponentiations in $\mathbb{G}_1, \mathbb{G}_2$ as well as modular multiplications and additions
- pairings and computations in $\mathbb{G}_T$ executed only by the terminal

Domain Signatures

Domain signatures
definition

System overview:

- a user holds one key in the chip (like for RI)
- many domains
- for each domain the user has a separate identity
- **for each domain the user creates signatures corresponding to his domain ID**

Motivation:

- RI is enough for authentication against a domain server
- . . . but sometimes the interaction with a domain requires **non-volatile authentication of the user's declarations**
- a regular signature is not really useful since:
  - the same public key used in different domain would link the identities
  - using a separate key pair for each domain would need a large number of keys and eID cards

Unforgeability:

- it is impossible to create a signature without the private key corresponding to the public key used for verification – the usual assumption!
- **but:** the adversary has potentially more data – the signatures of the same user with the same private key, but for different public keys of multiple domains
- **but:** a forgery is in particular changing a domain of a signature for a message *m*

Seclusiveness:

- **only a user with an eID issued by the system can create valid domain signatures**
- a generalization of PKI and certificates for regular signatures
- **but:** more complicated technically

a user asking for certificates for multiple domains at the same time would disclose the links between these domain identities and signatures

Desired properties
Unlinkability

Unlinkability:

- **impossible to link user's identities in different domains on input:**
  - public keys of some users in some domains
  - the corresponding signatures
  - for some users: links to public keys in all/some domains
  - private keys of some corrupted users
- the ideal situation: an adversary cannot distinguish two cases
  1. each uncorrupted user has public keys corresponding to a single private key
  2. each uncorrupted user has key pairs of chosen independently at random separately for each domain

Solution I - Jun Shao – M. Kutyłowski

Alice registers to a domain D

Input: domain D identity information $\mathtt{id}_D$

Alice secret key x_A

Output: public key $pk_{A,D}$ is registered in domain D

where $g_D = \mathrm{Hash}_1(\mathtt{id}_D)$ and $pk_{A,D} = g_D^{x_A}$

Alice creates a signature of m for domain D

$R = g_D^r$

$S = \mathrm{Hash}_2(g_D, pk_{A,D}, R, m) \cdot x_A + r \bmod q$

Output: signature $\sigma = (pk_{A,D}, R, S, m)$

a kind of Schnorr signature with domain specific generator

ShK - properties

Advantages:

- simplicity

Disadvantages:

- in each domain the user has to register explicitly in cooperation with the document issuer
- the user authenticates the domain public key with a proof of equality of discrete logarithms
- suited only for a small number of domains where
  - each user is in every domain
  - the issuing authority may learn the public keys of a user

ShK - properties
slight modification

Modified version

generation of domain generator g_D:

- Issuing Authority holds secret r_1
- domain D holds secret r_2
- $g_D = (g^{r_1})^{r_2}$

putting user's domain public key on the whitelist:

- Issuing Authority takes the main public key $pk = g^{x_A}$ of the user
- Issuing Authority computes $p_1 = pk^{r_1}$ and sends to the domain D
- domain D puts $p_2 = p_1^{r_2}$ on the whitelist

Computing the domain public key by the user

- fetch g_D
- compute $g_D^{x_A}$

the Issuing Authority does not know the domain public keys of the users

BSI algorithm
outline

- the original idea of domain signatures seems to originate from BSI
- the design influenced strongly by the legal limitations: the authorities are very limited to keep databases with citizens' personal data ($\Rightarrow$ no whitelists)
- published in

  J. Bender, J., Ö Dagdelen, K. Fischlin, D. Kügler: Domain-specific Pseudonymous Signatures for the German Identity Card. ISC'2012, LNCS 7483

  and indirectly referred to in

  Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token 2.20. BSI Technical Guideline TR-03110-2 (2015)

- the algorithm is based on Okamoto non-interactive proof of knowledge

BSI algorithm
core algorithm

Issuer's setup

- the secret keys z and x
- public keys $g_1, \quad g_2 = g_1^z, \quad y = g_1^x$

Issuing an eID for user i

- choose $x_{2,i} \in \mathbb{Z}_p$ at random
- compute $x_{1,i} = x - z \cdot x_{2,i}$
- install $(x_{1,i}, x_{2,i})$ in the eID of the user i.

Signing m by Alice for domain D

- create domain specific pseudonym $dsnym = D^{x_{1,i}}$
- choose t_1, t_2 at random, $a_1 = g_1^{t_1} g_2^{t_2}$, $a_2 = D^{t_1}$
- $c = \mathrm{Hash}(D, dsnym, a_1, a_2, m)$
- $s_1 = t_1 - c \cdot x_{i,1}$, $s_2 = t_2 - c \cdot x_{i,2}$
- output the signature (c, s_1, s_2)

Signing m by Alice for domain D

- create domain specific pseudonym $dsnym = D^{x_{1,i}}$
- choose t_1, t_2 at random, $a_1 = g_1^{t_1} g_2^{t_2}$, $a_2 = D^{t_1}$
- $c = \mathrm{Hash}(D, dsnym, a_1, a_2, m)$
- $s_1 = t_1 - c \cdot x_{i,1}$, $s_2 = t_2 - c \cdot x_{i,2}$
- output the signature (c, s_1, s_2)

Signature verification

- compute $a_1 = y^c \cdot g_1^{s_1} \cdot g_2^{s_2}$, $\quad a_2 = dsnym^c \cdot D^{s_1}$
- output `valid` if $c = \mathrm{Hash}(D, dsnym, a_1, a_2, m)$ and $dsnym$ not on a blacklist

BSI algorithm
verification justification

The values a_1, a_2 are reconstructed in a way analogous to Schnorr signatures:

$$
\begin{aligned}
y^c \cdot g_1^{s_1} \cdot g_2^{s_2} &= y^c \cdot g_1^{t_1 - c \cdot x_{i,1}} \cdot g_1^{z(t_2 - c \cdot x_{i,2})} \\
&= g_1^{t_1} \cdot g_2^{t_2} \cdot y^c \cdot g_1^{-c \cdot x_{i,1}} \cdot g_1^{-c \cdot z \cdot x_{i,2}} \\
&= a_1 \cdot y^c \cdot g_1^{-c \cdot (x_{i,1} + z \cdot x_{i,2})} = a_1 \cdot y^c \cdot g_1^{-c \cdot x} \\
&= a_1 \cdot y^c \cdot y^{-c} \\
&= a_1
\end{aligned}
$$

$$
\begin{aligned}
dsnym^c \cdot D^{s_1} &= D^{x_{i,1} \cdot c} \cdot D^{t_1 - c \cdot x_{i,1}} = D^{t_1} \\
&= a_2
\end{aligned}
$$

Advantages

lightweight infrastructure

Advantages:

- the main advantage of the scheme is that no certificate is required:
  a signature proves in fact that the signer knows $x_{i,1}, x_{i,2}$ such that $x = x_{1,i} + z \cdot x_{2,i}$
- no whitelist, certificates, ... needed, no limitation on the number of domains
- every user automatically in all domains

Seclusiveness problem

Attack:

- **break into just two eIDs**
- use private keys $x_{1,i}, x_{2,i}$ and $x_{1,j}, x_{2,j}$ to compute x, z based on the equations

$$
\begin{aligned}
x &= x_{1,i} + z \cdot x_{2,i} \\
x &= x_{1,j} + z \cdot x_{2,j}
\end{aligned}
$$

- ... and **create any number of fake eIDs** that would create proper domain signatures

only 1-seclusiveness holds, 2-seclusiveness does not hold

for a reliable implementation we need n-seclusiveness where n is a number of eIDs that a powerful adversary can acquire ($n \approx 10.000$?)

Unlinkability proof

- ill-designed unlinkability game
  - two pseudonyms
  - a signature corresponding to one of them
  - guess to which
- no correction in the IACR report despite of FC'2014 paper of French authors indicating the mistake

French domain signatures

- as an answer to seclusiveness problem of the BSI proposal
- published in

  J. Bringer, H. Chabanne, R. Lescuyer, A. Patey: Efficient and strongly secure dynamic domain-specific pseudonymous signatures for ID documents. Financial Cryptography 2014, LNCS 8437 and IACR Cryptology ePrint Archive 67 (2014)

French domain signatures
scheme, setup

Issuer's setup

- bilinear groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, of prime order p, bilinear mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with random generators $g_1, h \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$,
- secret key $\gamma \in \mathbb{Z}_p$, public key $y_1 = h^\gamma$, $y_2 = g_2^\gamma$

Issuing an eID for user *i* (some details omitted)

user *i* choose $f' \in \mathbb{Z}_p$ at random, $F' = h^{f'}$

user *i* send F' and a proof that it knows DL of F' to the Issuer

Issuer choose $x, f'' \in \mathbb{Z}_p$ at random, $F = F' \cdot h^{f''}$,

$$A = (g_1 \cdot F)^{1/(\gamma+x)}$$

Issuer send f'', A, x to the user

user *i* $f = f' + f''$, store (f, A, x) as the private key

Domain setup

- choose r at random
- $dpk = g_1^r$

User's domain specific pseudonym

- user's private key: (f, A, x)
- $nym = h^f \cdot dpk^x$

Signing m by user i for domain D

- user's private key: (f, A, x) , $Z = e(A, g_2)$
- pick $a, r_a, r_f, r_x, r_b, r_d \in \mathbb{Z}_p$ at random
- $T := A \cdot h^a$
- $R_1 := h^{r_f} \cdot dpk^{r_x}$
- $R_2 := nym^{r_a} \cdot h^{-r_d} \cdot dpk^{-r_b}$
- $R_3 := Z^{r_x} \cdot e(h, g_2)^{a \cdot r_x - r_f - r_b} \cdot e(h, y_2)^{-r_a}$
- $c := \mathrm{Hash}(dpk, nym, T, R_1, R_2, R_3, m)$
- $s_f := r_f + c \cdot f$, $s_x := r_x + c \cdot x$, $s_a := r_a + c \cdot a$, $s_b := r_b + c \cdot a \cdot x$; $s_d := r_d + c \cdot a \cdot f$
- Return $(T, c, s_f, s_x, s_a, s_b, s_d)$

French domain signatures

scheme - signature verification

Verifying a signature $(T, c, s_f, s_x, s_a, s_b, s_d)$ **for** m**,** nym **and** dpk

- $R_1 := h^{s_f} \cdot dpk^{s_x} \cdot nym^{-c}$
- $R_2 := nym^{s_a} \cdot h^{-s_d} \cdot dpk^{-s_b}$
- $R_3 := e(T, g_2)^{s_x} \cdot e(h, g_2)^{-s_f - s_b} \cdot e(h, y_2)^{-s_a} \cdot \left(e(g_1, g_2) \cdot e(T, y_2)\right)^{-c}$
- output `valid` if $c = \mathrm{Hash}(dpk, nym, T, R_1, R_2, R_3, m)$

French domain signatures

remarks

Advantages

- breaking into some number of eID's does not enable to create fake users – just as needed in the practical scenario
- some additional mechanisms for user revocation

Disadvantages

- complicated, unclear for human inspection (security risk)
- problems with security model
- computational complexity – (too) heavy for smart cards

description of oracles of the security model:

$\mathtt{AddDomain}(j)$
- if $j \in \mathcal{D}$, then abort
- $\mathbf{RL}[j] := \{\}$; $\mathtt{All}[j] := copy(\mathcal{HU})$
- $\mathbf{dpk}[j] \leftarrow \mathtt{DomainKeyGen}(\mathsf{gpk}, j)$
- $\forall i \in \mathcal{HU}$,
  - $\mathbf{\Sigma}[(i,j)] := \{\}$; $\mathsf{UU}[(i,j)] := \&(\mathtt{All}[j])$
  - $\mathbf{nym}[i][j] \leftarrow \mathtt{NymGen}(\mathsf{gpk}, \mathbf{dpk}[j], \mathbf{usk}[i])$
- return $\mathbf{dpk}[j]$

$\mathtt{CorruptUser}(i)$
- if $i \in \mathcal{HU} \cup \mathcal{CU}$, then abort
- $\mathcal{CU} := \mathcal{CU} \cup \{i\}$
- $\mathbf{usk}[i] := \perp$; $\mathbf{nym}[i] := \perp$; $\mathbf{rt}[i] := \perp$
- $\mathsf{dec}[IA][i] := \mathsf{cont}$; $\mathsf{state}[IA][i] := (\mathsf{gpk}, \mathsf{isk})$

$\mathtt{Nym}(i,j)$
- if $i \notin \mathcal{HU}$ or $j \notin \mathcal{D}$ or $(i,j) \in \mathcal{CH}$, abort
- $\mathsf{UU}[(i,j)] := \{i\}$; $\mathtt{All}[j] := \mathtt{All}[j] \setminus \{i\}$
- $\forall i' \in \mathcal{HU} \setminus \{i\}$, if $\mathsf{UU}[(i',j)] \neq \&(\mathtt{All}[j])$,
  then $\mathsf{UU}[(i',j)] := \mathsf{UU}[(i',j)] \setminus \{i\}$
- return $\mathbf{nym}[i][j]$

$\mathtt{NymDomain}(j)$
- if $j \notin \mathcal{D}$, then abort
- $result := random\_perm(copy(\mathtt{All}[j]))$
- $\forall i \in \mathcal{HU}$,
  - if $\mathsf{UU}[(i,j)] == \&(\mathtt{All}[j])$,
    - $\mathsf{UU}[(i,j)] := copy(\mathtt{All}[j])$
- $\mathtt{All}[j] := \{\}$; return $\{\mathbf{nym}[i][j]\}_{i \in result}$

$\mathtt{Sign}(i,j,m)$
- if $i \notin \mathcal{HU}$ or $j \notin \mathcal{D}$, then abort
- $\mathbf{\Sigma}[(i,j)] := \mathbf{\Sigma}[(i,j)] \cup \{m\}$
- return $\mathtt{Sign}(\mathsf{gpk}, \mathbf{dpk}[j], \mathbf{usk}[i], \mathbf{nym}[i][j], m)$

$\mathtt{ReadRegistrationTable}(i)$
- return $\mathbf{rt}[i]$

$\mathtt{WriteRegistrationTable}(i, M)$
- $\mathbf{rt}[i] := M$

$\mathtt{AddUser}(i)$
- if $i \in \mathcal{HU} \cup \mathcal{CU}$, then abort
- $\mathcal{HU} := \mathcal{HU} \cup \{i\}$
- run $\mathsf{usk} \leftarrow \mathtt{Join}(\mathsf{gpk}) \leftrightarrow \mathtt{Issue}(\mathsf{gpk}, \mathsf{isk}) \rightarrow \mathsf{rt}$
- $\mathbf{usk}[i] := \mathsf{usk}$; $\mathbf{rt}[i] := \mathsf{rt}$
- $\forall j \in \mathcal{D}$,
  - $\mathbf{\Sigma}[(i,j)] := \{\}$; $\mathtt{All}[j] := \mathtt{All}[j] \cup \{i\}$
  - $\mathbf{nym}[i][j] \leftarrow \mathtt{NymGen}(\mathsf{gpk}, \mathbf{dpk}[j], \mathbf{usk}[i])$
  - $\mathsf{UU}[(i,j)] := \&(\mathtt{All}[j])$

$\mathtt{UserSecretKey}(i)$
- if $i \notin \mathcal{HU}$ or $\exists j \in \mathcal{D}$, s.t. $(i,j) \in \mathcal{CH}$, abort
- $\mathcal{HU} := \mathcal{HU} \setminus \{i\}$; $\mathcal{CU} := \mathcal{CU} \cup \{i\}$
- $\forall j \in \mathcal{D}$,
  - $\mathsf{UU}[(i,j)] := \{i\}$; $\mathtt{All}[j] := \mathtt{All}[j] \setminus \{i\}$
  - $\forall i' \in \mathcal{HU}$, if $\mathsf{UU}[(i',j)] \neq \&(\mathtt{All}[j])$,
    then $\mathsf{UU}[(i',j)] := \mathsf{UU}[(i',j)] \setminus \{i\}$
- return $(\mathbf{usk}[i], \mathbf{nym}[i])$

$\mathtt{Revoke}(i, \mathcal{D}')$
- $\forall j \in \mathcal{D}'$, call $\mathtt{DomainRevoke}(i,j)$
- return $\{\mathbf{RL}[j]\}_{j \in \mathcal{D}'}$

$\mathtt{DomainRevoke}(i,j)$
- if $i \notin \mathcal{HU}$ or $j \notin \mathcal{D}$ or $(i,j) \in \mathcal{CH}$, then abort
- $\mathsf{aux} \leftarrow \mathtt{Revoke}(\mathsf{gpk}, \mathbf{rt}[i], \{\mathbf{dpk}[j]\})$
- $\mathbf{RL}[j] \leftarrow \mathtt{DomainRevoke}(\mathbf{dpk}[j], \mathsf{aux}, \mathbf{RL}[j])$
- $\mathsf{UU}[(i,j)] := \{i\}$; $\mathtt{All}[j] := \mathtt{All}[j] \setminus \{i\}$
- $\forall i' \in \mathcal{HU} \setminus \{i\}$, if $\mathsf{UU}[(i',j)] \neq \&(\mathtt{All}[j])$,
  then $\mathsf{UU}[(i',j)] := \mathsf{UU}[(i',j)] \setminus \{i\}$
- return $\mathbf{RL}[j]$

$\mathtt{NymSign}(\mathsf{nym}, j, m)$
- if $j \notin \mathcal{D}$, then abort
- find $i \in \mathcal{HU}$ such that $\mathbf{nym}[i][j] == \mathsf{nym}$
  if no match is found, then abort
- $\mathbf{\Sigma}[(i,j)] := \mathbf{\Sigma}[(i,j)] \cup \{m\}$
- return $\mathtt{Sign}(\mathsf{gpk}, \mathbf{dpk}[j], \mathbf{usk}[i], \mathbf{nym}[i][j], m)$

$\mathtt{SendToUser}(i, M_{in})$
- if $i \in \mathcal{CU}$, then abort ; if $i \notin \mathcal{HU}$, then
  $\mathcal{HU} := \mathcal{HU} \cup \{i\}$; $M_{in} := \varepsilon$; $\mathbf{usk}[i] := \perp$; $\mathsf{state}[i][IA] := \mathsf{gpk}$; $\mathsf{dec}[i][IA] := \mathsf{cont}$
- $(\mathsf{state}[i][IA], M_{out}, \mathsf{dec}[i][IA]) \leftarrow \mathtt{Join}(\mathsf{state}[i][IA], M_{in}, \mathsf{dec}[i][IA])$
- if $\mathsf{dec}[i][IA] == \mathsf{accept}$, then $\mathbf{usk}[i] := \mathsf{state}[i][IA]$
- return $(M_{out}, \mathsf{dec}[i][IA])$

$\mathtt{SendToIssuer}(i, M_{in})$
- if $i \notin \mathcal{CU}$, then abort
- $(\mathsf{state}[IA][i], M_{out}, \mathsf{dec}[IA][i]) \leftarrow \mathrm{DSPS}.\mathtt{Issue}(\mathsf{state}[IA][i], M_{in}, \mathsf{dec}[IA][i])$
- if $\mathsf{dec}[IA][i] == \mathsf{accept}$, then set $\mathbf{rt}[i] := \mathsf{state}[IA][i]$
- return $(M_{out}, \mathsf{dec}[IA][i])$

$\mathtt{Challenge}(b_A, b_B, j_A, j_B, i_0, i_1)$
- if $i_0 \notin \mathcal{HU}$ or $i_1 \notin \mathcal{HU}$ or $i_0 == i_1$ or $j_A \notin \mathcal{D}$ or $j_B \notin \mathcal{D}$ or $j_A == j_B$, then abort
- if $\forall j \in \{j_A, j_B\}$, $\exists i \in \{i_0, i_1\}$ such that $\{i_0, i_1\} \not\subseteq \mathsf{UU}[(i,j)]$, then abort
- $\mathcal{CH} := \{(i_0, j_A), (i_0, j_B), (i_1, j_A), (i_1, j_B)\}$; return $(\mathbf{nym}[i_{b_A}][j_A], \mathbf{nym}[i_{b_B}][j_B])$

Figure 1: Oracles provided to adversaries

8

French domain signatures
security proof

static model:

- the users created in advance,
- the set of corrupted users determined in advance,

static versus dynamic adversary

despite the declarations:
security proofs do not fully cover the dynamic model, where the adversary may adaptively corrupt the users
some additional assumptions hidden in order to pass the proofs

French domain signatures
Delegation - key leakage

- for decreasing the complexity of computation of eID, computations delegated to the PC operating the reader
- two different methods of delegation (FC paper, a more efficient one in the IACR report)

Citation from FC paper: *"In our construction,* ***the adversary can compute*** *A from B_2 and σ (if $\sigma = (T, c, s_f, s_x, s_a, s_b, s_d)$, then $A = T \cdot (B_2 \cdot h^{s_a})^{-1/c}$. The fact that we can simulate signatures even in the cross-domain anonymity game shows that the knowledge of A does not help linking users across domains."*

key leakage

A, a part of the secret key is leaked to the PC

identity leakage

the PC may link the pseudonyms of the same eID in different domains via A

French domain signatures

further issues

some further issues concerning incompleteness of security proofs will be published in Kamil Kluczniak PhD Dissertation

the scheme seems to require a lot of attention, some modifications and surely a careful proofreading before one can talk about readiness for a practical deployment

Kluczniak's domain signature schemes

to appear in

Kamil Kluczniak, Anonymous Authentication Using Electronic Identity Documents, PhD Dissertation to be submitted at Polish Academy of Science

Kluczniak's domain signatures - other schemes

- altogether 4 schemes proposed
- tradeoff between simplicity of the scheme and strength of the adversary model
- one of the schemes has neither pairings nor exponentiations in $\mathbb{G}_T$
- all schemes are Sigma-protocols and therefore can be converted to Restricted Identification
- two schemes are provably secure in the dynamic model

Corollaries

Lesson learnt

- many new concepts
- ... but at the same time a lot of problems
- **cryptographic algorithms of fundamental importance for privacy protection deployed without much inspection by independent cryptographic community**

Thanks for your attention!

Contact data

1. Miroslaw.Kutylowski@pwr.edu.pl
2. http://kutylowski.im.pwr.edu.pl

Supported by Foundation for Polish Science: MISTRZ and VENTURES programs