Ad-Hoc-Domain Signatures for Personal eID Documents

Kamil Kluczniak, Lucjan Hanzlik, Mirosław Kutyłowski

Wrocław University of Science and Technology, Poland

ArcticCrypt 2016,
Longyearbyen, Svalbard

Overview

Motivation

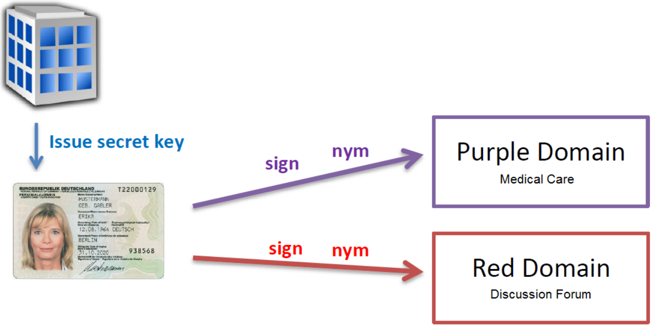
- **eIDAS - EU REGULATION No 910/2014**
  identification, authentication and other trust services in the European market
- **growing scope of usage of electronic documents**
  reliable authentication of documents badly needed. Electronic signatures one of a few reliable choices.
- **“Privacy by Design” paradigm**
  a technical system must be designed in a way that protects privacy
  privacy protection is a fundamental security condition

Domain Pseudonyms Concept

Pseudonym:

A unique ID in each service that does not reveal the real identity

preventing Sybil attacks: appearing under different IDs in the same service.



Domain Pseudonyms Signatures Concept

Domain Signatures:

1. one user - just one private key for all domains

Domain Pseudonyms Signatures Concept

Domain Signatures:

1. one user - just one private key for all domains
2. domain pseudonym acts as a public key

Domain Pseudonyms Signatures Concept

Domain Signatures:

1. one user - just one private key for all domains
2. domain pseudonym acts as a public key
3. verification related to the domain pseudonym

Domain Pseudonyms Signatures Concept

Domain Signatures:

1. one user - just one private key for all domains
2. domain pseudonym acts as a public key
3. verification related to the domain pseudonym
4. verification must not reveal the real identity

Domains and Requirements

Domain/Sector

Service area where the user must appear under the same (pseudonymous) identity.

like a user account

Domains and Requirements

Domain/Sector

Service area where the user must appear under the same (pseudonymous) identity.

like a user account

Unlinkability

The pseudonyms in different sectors must be unlinkable.

Domains and Requirements

Domain/Sector

Service area where the user must appear under the same (pseudonymous) identity.

like a user account

Unlinkability

The pseudonyms in different sectors must be unlinkable.

Seclusiveness

Only the Issuer may create/admit new users.

like for issuing personal ID cards

Requirements

Revocation

The Issuer can revoke a user within a domain.

like for stolen personal ID cards

Requirements

Revocation

The Issuer can revoke a user within a domain.

like for stolen personal ID cards

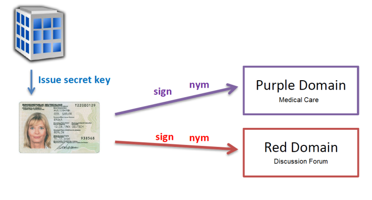
Pseudonym Uniqueness - Resistance to Sybil attacks

A user may have just one pseudonym per domain.

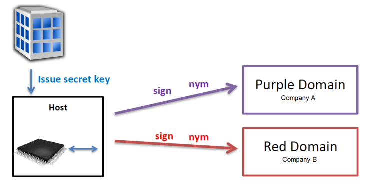
previous work was focused on this, but surprisingly a formal requirement was missing

Comparison to Direct Anonymous Attestation

Domain Signatures



Direct Anonymous Attestation



Comparison to Direct Anonymous Attestation

	Ad Hoc DS:	**DAA:**
Environment:	Smart Cards	Host with TPM
Privacy issues:	a reader is a privacy threat	host is **NOT** a privacy threat
Revocation method:	blacklist a pseudonym	publish the secret key
Updating the state of a device:	Impossible	Possible

Comparison to Direct Anonymous Attestation

	Ad Hoc DS:	**DAA:**
Environment:	Smart Cards	Host with TPM
Privacy issues:	a reader is a privacy threat	host is **NOT** a privacy threat
Revocation method:	blacklist a pseudonym	publish the secret key
Updating the state of a device:	Impossible	Possible

- differences mainly implied by the execution environment

Comparison to Direct Anonymous Attestation

	Ad Hoc DS:	**DAA:**
Environment:	Smart Cards	Host with TPM
Privacy issues:	a reader is a privacy threat	host is **NOT** a privacy threat
Revocation method:	blacklist a pseudonym	publish the secret key
Updating the state of a device:	Impossible	Possible

- differences mainly implied by the execution environment
- in contrast to Domain Signatures, DAA does not have a revocation method without publishing the secret key

Domain Signatures - Formal Definition

Procedures

Setup: $\mathrm{Setup}(1^k) \rightarrow (gPK, iSK)$

Domain Signatures - Formal Definition

Procedures

Setup: $\mathrm{Setup}(1^k) \to (gPK, iSK)$

Join/Issue: $(uSK[i]) \leftarrow \mathrm{Join}(gPK, i) \leftrightarrow \mathrm{Issue}(gPK, iSK, uRT) \to (uRT[i])$

Domain Signatures - Formal Definition

Procedures

Setup: $\mathrm{Setup}(1^k) \rightarrow (gPK, iSK)$

Join/Issue: $(uSK[i]) \leftarrow \mathrm{Join}(gPK, i) \leftrightarrow \mathrm{Issue}(gPK, iSK, uRT) \rightarrow (uRT[i])$

Generate Pseudonym: $\mathrm{NymGen}(gPK, \mathtt{dom}, uSK[i]) \rightarrow nym$

Domain Signatures - Formal Definition

Procedures

Setup: $\mathrm{Setup}(1^k) \rightarrow (gPK, iSK)$

Join/Issue: $(uSK[i]) \leftarrow \mathrm{Join}(gPK, i) \leftrightarrow \mathrm{Issue}(gPK, iSK, uRT) \rightarrow (uRT[i])$

Generate Pseudonym: $\mathrm{NymGen}(gPK, \mathtt{dom}, uSK[i]) \rightarrow nym$

Generate Domain Revocation Token:
$\mathrm{DomainRevocationTokenGen}(gPK, \mathtt{dom}, uRT[i]) \rightarrow dRT[i]$

Domain Signatures - Formal Definition

Procedures

Setup: $\mathrm{Setup}(1^k) \to (gPK, iSK)$

Join/Issue: $(uSK[i]) \leftarrow \mathrm{Join}(gPK, i) \leftrightarrow \mathrm{Issue}(gPK, iSK, uRT) \to (uRT[i])$

Generate Pseudonym: $\mathrm{NymGen}(gPK, \mathtt{dom}, uSK[i]) \to nym$

Generate Domain Revocation Token:
$\mathrm{DomainRevocationTokenGen}(gPK, \mathtt{dom}, uRT[i]) \to dRT[i]$

Revocation Check: $\mathrm{RevocationCheck}(dPK, \mathtt{dom}, nym, dRT[i]) \to \{0, 1\}$

Domain Signatures - Formal Definition

Procedures

Setup: $\mathrm{Setup}(1^k) \to (gPK, iSK)$

Join/Issue: $(uSK[i]) \leftarrow \mathrm{Join}(gPK, i) \leftrightarrow \mathrm{Issue}(gPK, iSK, uRT) \to (uRT[i])$

Generate Pseudonym: $\mathrm{NymGen}(gPK, \mathtt{dom}, uSK[i]) \to nym$

Generate Domain Revocation Token: $\mathrm{DomainRevocationTokenGen}(gPK, \mathtt{dom}, uRT[i]) \to dRT[i]$

Revocation Check: $\mathrm{RevocationCheck}(dPK, \mathtt{dom}, nym, dRT[i]) \to \{0, 1\}$

Sign: $\mathrm{Sign}(gPK, \mathtt{dom}, uSK[i], m) \to \sigma$

Domain Signatures - Formal Definition

Procedures

Setup: $\mathrm{Setup}(1^k) \to (gPK, iSK)$

Join/Issue: $(uSK[i]) \leftarrow \mathrm{Join}(gPK, i) \leftrightarrow \mathrm{Issue}(gPK, iSK, uRT) \to (uRT[i])$

Generate Pseudonym: $\mathrm{NymGen}(gPK, \mathtt{dom}, uSK[i]) \to nym$

Generate Domain Revocation Token: $\mathrm{DomainRevocationTokenGen}(gPK, \mathtt{dom}, uRT[i]) \to dRT[i]$

Revocation Check: $\mathrm{RevocationCheck}(dPK, \mathtt{dom}, nym, dRT[i]) \to \{0, 1\}$

Sign: $\mathrm{Sign}(gPK, \mathtt{dom}, uSK[i], m) \to \sigma$

Verify: $\mathrm{Verify}(gPK, \mathtt{dom}, nym, m, \sigma) \to \{0, 1\}$:

Unforgeability

1. The adversary obtains Issuer’s secret key

Unforgeability

1. The adversary obtains Issuer's secret key
2. The adversary may:
   - add new honest users – as the Issuer,
   - ask for pseudonyms, signatures and user secret keys.

Unforgeability

1. The adversary obtains Issuer's secret key
2. The adversary may:
   - add new honest users – as the Issuer,
   - ask for pseudonyms, signatures and user secret keys.
3. The adversary returns a pseudonym *nym*, a domain *dom* and a signature σ on message *m*, and wins if:

Unforgeability

1. The adversary obtains Issuer's secret key
2. The adversary may:
   - add new honest users – as the Issuer,
   - ask for pseudonyms, signatures and user secret keys.
3. The adversary returns a pseudonym *nym*, a domain *dom* and a signature σ on message *m*, and wins if:
   - The signature σ verifies correctly with respect to *nym* and *dom*

Unforgeability

1. The adversary obtains Issuer's secret key
2. The adversary may:
   - add new honest users – as the Issuer,
   - ask for pseudonyms, signatures and user secret keys.
3. The adversary returns a pseudonym *nym*, a domain *dom* and a signature σ on message *m*, and wins if:
   - The signature σ verifies correctly with respect to *nym* and *dom*
   - The revocation token of some user *i* revokes *nym*.

Unforgeability

1. The adversary obtains Issuer's secret key
2. The adversary may:
   - add new honest users – as the Issuer,
   - ask for pseudonyms, signatures and user secret keys.
3. The adversary returns a pseudonym *nym*, a domain *dom* and a signature σ on message *m*, and wins if:
   - The signature σ verifies correctly with respect to *nym* and *dom*
   - The revocation token of some user *i* revokes *nym*.
   - The adversary has not asked for the secret key of this user.

Seclusiveness

1. The adversary creates all users by interacting with the Issuer.
   (all users are under control of the adversary)

Seclusiveness

1. The adversary creates all users by interacting with the Issuer.
   (all users are under control of the adversary)
2. The adversary returns a pseudonym *nym*, a domain *dom* and a signature σ on a message m.

Seclusiveness

1. The adversary creates all users by interacting with the Issuer.
   (all users are under control of the adversary)
2. The adversary returns a pseudonym *nym*, a domain *dom* and a signature σ on a message *m*.
3. The adversary , and wins if:
   - The signature σ verifies correctly with respect to *nym* and *dom*.

Seclusiveness

1. The adversary creates all users by interacting with the Issuer.
   (all users are under control of the adversary)
2. The adversary returns a pseudonym *nym*, a domain *dom* and a signature σ on a message *m*.
3. The adversary , and wins if:
   - The signature σ verifies correctly with respect to *nym* and *dom*.
   - No revocation token created by the Issuer revokes *nym*.

Pseudonym Uniqueness

1. The adversary obtains the Issuer's secret key.

Pseudonym Uniqueness

1. The adversary obtains the Issuer's secret key.
2. His goal is to return a revocation token uRT, a domain dom, and tuples (m_0, nym_0, σ_0) and (m_1, nym_1, σ_1).

Pseudonym Uniqueness

1. The adversary obtains the Issuer's secret key.
2. His goal is to return a revocation token uRT, a domain dom, and tuples (m_0, nym_0, σ_0) and (m_1, nym_1, σ_1).
3. The adversary wins if
   - signatures σ_0, σ_1 verify correctly with respect to (m_0, nym_0) and (m_1, nym_1), respectively,

Pseudonym Uniqueness

1. The adversary obtains the Issuer's secret key.
2. His goal is to return a revocation token uRT, a domain dom, and tuples (m_0, nym_0, σ_0) and (m_1, nym_1, σ_1).
3. The adversary wins if
   - signatures σ_0, σ_1 verify correctly with respect to (m_0, nym_0) and (m_1, nym_1), respectively,
   - uRT revokes both nym_0 and nym_1.

Note - Identification of a User

- Note that in each experiment, **the challenger identifies the signer** (or may identify that no such signer exist).

Note - Identification of a User

- Note that in each experiment, **the challenger identifies the signer** (or may identify that no such signer exist).
- In Direct Anonymous Attestation the challenger cannot identify the signer...

Note - Identification of a User

- Note that in each experiment, **the challenger identifies the signer** (or may identify that no such signer exist).
- In Direct Anonymous Attestation the challenger cannot identify the signer...
- In DAA challenger does not even know, **whether the adversary broke unforgeability or seclusiveness.**

Note - Identification of a User

- Note that in each experiment, **the challenger identifies the signer** (or may identify that no such signer exist).
- In Direct Anonymous Attestation the challenger cannot identify the signer...
- In DAA challenger does not even know, **whether the adversary broke unforgeability or seclusiveness.**
- **In the security proofs for DAA**, establishing the origin of the signature is done by an artificial procedure (e.g. knowledge extractor in ROM).

Unlinkability - Game Based Definition

- We may assign an index to every user in the system.

Unlinkability - Game Based Definition

- We may assign an index to every user in the system.
- The adversary may ask for,
  - pseudonyms signatures and private keys of the *i*th user,

Unlinkability - Game Based Definition

- We may assign an index to every user in the system.
- The adversary may ask for,
  - pseudonyms signatures and private keys of the ith user,

If the adversary gives as input user indexes, he knows exactly which pseudonyms belong to which users.

Unlinkability - Game Based Definition

- We may assign an index to every user in the system.
- The adversary may ask for,
  - pseudonyms signatures and private keys of the ith user,

If the adversary gives as input user indexes, he knows exactly which pseudonyms belong to which users.

Example

- Pseudonym of the i-th user in domain $dom_1 \rightarrow nym_1$
- Pseudonym of the i-th user in domain $dom_2 \rightarrow nym_2$

Unlinkability - Previous work

Game based definitions

- Bender, Dagdelen, Fischlin, Kügler: ISC 2012 [BDFK12]
  - a mistake, every adversary can win the game.

Unlinkability - Previous work

Game based definitions

- Bender, Dagdelen, Fischlin, Kügler: ISC 2012 [BDFK12]

  - a mistake, every adversary can win the game.

- Bringer, Chabanne, Lescuyer, Patey: Financial Cryptography 2014 [BCLP14]
  - attempt to cover the problem with "uncertainty sets"
  - obscure and hard to understand
  - restricts the adversary to some narrow strategies and does not cover some real world cases

Unlinkability - Previous work

Game based definitions

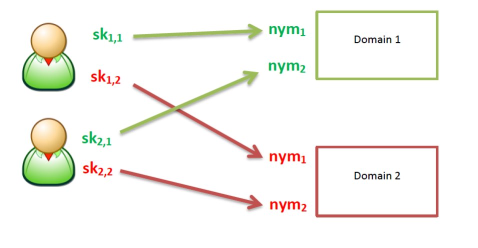
- Bender, Dagdelen, Fischlin, Kügler: ISC 2012 [BDFK12]

  - a mistake, every adversary can win the game.
- Bringer, Chabanne, Lescuyer, Patey: Financial Cryptography 2014 [BCLP14]
  - attempt to cover the problem with “uncertainty sets”
  - obscure and hard to understand
  - restricts the adversary to some narrow strategies and does not cover some real world cases
- Brickell, Chen, Li: International Journal of Information Security [BCL09]

  - considers just two users in one domain.

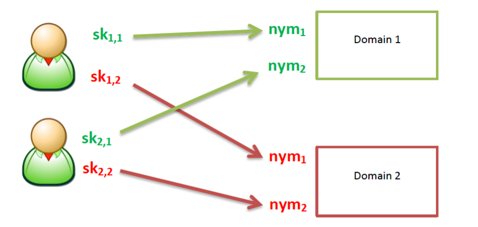
Change of concept for Defining Unlinkability

Ideal World

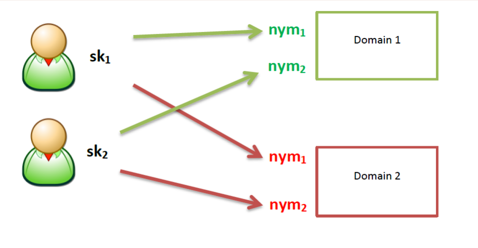


Change of concept for Defining Unlinkability

Ideal World



Real World



Defining unlinkability

long story of problems with a formal treatment

Two approaches

- Game Based definitions - huge problems for pseudonym unlinkability
- Simulation based approaches - static corruptions only

Defining unlinkability
long story of problems with a formal treatment

Two approaches

- Game Based definitions - huge problems for pseudonym unlinkability
- Simulation based approaches - static corruptions only

New approaches

- this work - game based definitions, except for anonymity which is simulation based:
  how much new knowledge for the adversary is brought by the particular crypto algorithm instead of independent keys for each domain

Defining unlinkability

long story of problems with a formal treatment

Two approaches

- Game Based definitions - huge problems for pseudonym unlinkability
- Simulation based approaches - static corruptions only

New approaches

- this work - game based definitions, except for anonymity which is simulation based:
  how much new knowledge for the adversary is brought by the particular crypto algorithm instead of independent keys for each domain
- Camenisch, Drijver, Lehmann: "Universally Composable Direct Anonymous Attestation" - via UC Framework.

Designs related to Pseudonymous Signature

1. prototype of PS: [BDFK12] Bender, Dagdelen, Fischlin, Kügler: ISC 2012
   - No seclusiveness. If the adversary gets two secret key, then he might compute the Issuer's secret key

Designs related to Pseudonymous Signature

1. prototype of PS: [BDFK12] Bender, Dagdelen, Fischlin, Kügler: ISC 2012
   - No seclusiveness. If the adversary gets two secret key, then he might compute the Issuer's secret key
2. a solution from pairings but no group key problem: [BCLP14] Bringer, Chabanne, Lescuyer, Patey: Financial Cryptography 2014
   - Minor problems (proofs do not work).
   - Pairing delegation procedure leaks partially the user's secret key.

Designs related to Pseudonymous Signature

1. prototype of PS: [BDFK12] Bender, Dagdelen, Fischlin, Kügler: ISC 2012
   - No seclusiveness. If the adversary gets two secret key, then he might compute the Issuer's secret key
2. a solution from pairings but no group key problem: [BCLP14] Bringer, Chabanne, Lescuyer, Patey: Financial Cryptography 2014
   - Minor problems (proofs do not work).
   - Pairing delegation procedure leaks partially the user's secret key.
3. solution from pairings, model issues fixed: this work

Ad hoc Domain Signatures

Solution Overview

- Boneh-Boyen like signature based on user's secret key:
$$(u, x, A = (g \cdot h^{x})^{1/(z+u)})$$

Ad hoc Domain Signatures

Solution Overview

- Boneh-Boyen like signature based on user's secret key:

$$(u, x, A = (g \cdot h^{x})^{1/(z+u)})$$

- deriving a pseudonym of a user in a domain

$$nym = \text{Hash}(\texttt{domain-name})^{u} \cdot g^{x}$$

Ad hoc Domain Signatures

Solution Overview

- Boneh-Boyen like signature based on user's secret key:
$$(u, x, A = (g \cdot h^x)^{1/(z+u)})$$
- deriving a pseudonym of a user in a domain
$$nym = \mathrm{Hash}(\texttt{domain-name})^u \cdot g^x$$
- Signing via a Sigma Protocol and Fiat-Shamir transformation:

$$ZKPoK\{(\alpha, \beta, \gamma) :$$
$$nym = \mathrm{H}(\texttt{domain-name})^\alpha \cdot g^\beta \wedge \gamma^{z+\alpha} \cdot h^{-\beta} = g_1\}$$

Efficiency comparison

Signature Size

Scheme	$\mathbb{G}_1$	$\mathbb{G}_2$	$\mathbb{G}_T$	$\mathbb{Z}_q$	Bit Size[1]
Our scheme	1	0	0	6	1792
[BDFK12]	0	0	0	3	768
[BCLP14]	1	0	0	6	1792

Signature Creation

Scheme	Multiplications	Exponentiations
Our Scheme	$3 \cdot \mathbb{G}_1 + 2 \cdot \mathbb{G}_T$	$6 \cdot \mathbb{G}_1 + 3 \cdot \mathbb{G}_T$
[BDFK12]	$1 \cdot \mathbb{G}_1$	$3 \cdot \mathbb{G}_1$
[BCLP14]	$4 \cdot \mathbb{G}_1 + 2 \cdot \mathbb{G}_T$	$6 \cdot \mathbb{G}_1 + 3 \cdot \mathbb{G}_T$

Signature Verification

Scheme	Multiplications	Exponentiations	Inv.	Pairing
Our Scheme	$4 \cdot \mathbb{G}_1 + 1 \cdot \mathbb{G}_2 + 2 \cdot \mathbb{G}_T$	$6 \cdot \mathbb{G}_1 + 2 \cdot \mathbb{G}_2 + 2 \cdot \mathbb{G}_T$	0	1
[BDFK12]	$1 \cdot \mathbb{G}_1$	$3 \cdot \mathbb{G}_1$	0	0
[BCLP14]	$4 \cdot \mathbb{G}_1 + 2 \cdot \mathbb{G}_T$	$6 \cdot \mathbb{G}_1 + 3 \cdot \mathbb{G}_T$	$1 \cdot \mathbb{G}_T$	2

[1] Counted according to RFC3766 for 256-bit representation $\mathbb{Z}_p$, $\mathbb{G}_1$ and 512-bit $\mathbb{G}_2$. (3707-bit RSA modulus)

Open Problems - Revocation

- The current state-of-the-art:
  we may:
  - request a signer to update his state (download new credentials/certificates), or
  - use blacklists like in VRL Group Signatures.

Open Problems - Revocation

- The current state-of-the-art:
  we may:
  - request a signer to update his state (download new credentials/certificates), or
  - use blacklists like in VRL Group Signatures.
- If there are blacklists, then a the party which creates blacklists (issuer) may trace users.

Open Problems - Revocation

- The current state-of-the-art:
  we may:
  - request a signer to update his state (download new credentials/certificates), or
  - use blacklists like in VRL Group Signatures.
- If there are blacklists, then a the party which creates blacklists (issuer) may trace users.
- For Ad Hoc Domain Signatures: we may not be aware about every domain used, thus it is hard to blacklist.

Conclusions

- We gave a new and presumably correct definition for Ad Hoc Domain Signatures.
  At least some issues from previous works are solved.

Conclusions

- We gave a new and presumably correct definition for Ad Hoc Domain Signatures.
  At least some issues from previous works are solved.
- It may prove useful for giving a sound definition for Direct Anonymous Attestation.

Conclusions

- We gave a new and presumably correct definition for Ad Hoc Domain Signatures.
  At least some issues from previous works are solved.
- It may prove useful for giving a sound definition for Direct Anonymous Attestation.
- We designed an "efficient" (?) scheme from Bilinear Maps.

Conclusions

- We gave a new and presumably correct definition for Ad Hoc Domain Signatures.
  At least some issues from previous works are solved.
- It may prove useful for giving a sound definition for Direct Anonymous Attestation.
- We designed an “efficient” (?) scheme from Bilinear Maps.
- Revocation may still be a problem.

Thank You