Rethinking Identification Protocols from the Point of View of the GDPR

Mirosław Kutyłowski[1,2], Lukasz Krzywiecki[1], Xiaofeng Chen[2]

[1] Wrocław University of Science and Technology, Wrocław, Poland
[2] Xidian University, Xi'an, P.R. China

CSCML 2019, Be'er Sheva

Identification between electronic artefacts

M. Kutyłowski, L. Krzywiecki, X. Chen

Actors

Verifier: checks identity of the Prover

Prover: authenticates itself against the Verifier

Mechanism

the Prover convinces the Verifier that it holds the private key assigned to the Prover:

- the right key is used $\Rightarrow$ verification succeeds
- a wrong key used $\Rightarrow$ verification succeeds with a negligible probability

Identification between electronic artefacts
the simplest method

M. Kutyłowski, L. Krzywiecki, X. Chen

Protocol

we assume that the Verifier knows the public key of the Prover

1. the Verifier generates a random challenge *r* and sends it to the Prover
2. the Prover creates a signature *s* of *r* and returns it to the Verifier
3. the Verifier checks the signature *s*

what is wrong with it?

such a protocol provides a stronger proof than required

if *r* is a signature of the Verifier, then *s* becomes an undeniable proof for a third party that the Prover has interacted with the Verifier

GDPR versus cryptographic protocols

M. Kutyłowski, L. Krzywiecki, X. Chen

personal data

'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly

an artefact and its activity may be related to a natural person

by definition, identification protocol provides information relating to an identified participant

GDPR versus cryptographic protocols

Data minimality principle

a system should not gather more data than it is necessary to achieve its purpose.

Motivation

more data $\Rightarrow$ more risks:
an intruder gains more data and can misuse it for malicious purposes.

Consequence

If it is possible to achieve a purpose without processing data D, then processing D is unlawful.
(by definition, creating D is a kind of data processing)

GDPR versus cryptographic protocols

M. Kutyłowski,
L. Krzywiecki,
X. Chen

Purpose limitation principle

"personal data shall be collected for specified, explicit and legitimate purposes and not further processed in a manner that is incompatible with those purposes"

Problems

if data created and transmitted over a wireless channel, then anybody can further process it in an arbitrary way

strong cryptographic proofs - like digital signatures - facilitate "further processing" due to origin and integrity guarantees

GDPR versus cryptographic protocols

M. Kutyłowski,
L. Krzywiecki,
X. Chen

Storage limitation

data *“kept in a form which permits identification of data subjects for no longer than is necessary for the purposes for which the personal data are processed”*

Problems

If identification runs in public, then it is infeasible to ensure that the observers will forget the identification data.

GDPR versus cryptographic protocols

Integrity and confidentiality

personal data shall be processed in a manner that ensures appropriate security of the personal data, including protection against unauthorized or unlawful processing [. . .] using appropriate technical or organizational measures.

Consequences

- “appropriate security”
  $\Rightarrow$ risk analysis
- based on “technical or organizational measures” and not on compensation
  $\Rightarrow$ privacy by design

GDPR versus cryptographic protocols

M. Kutyłowski,
L. Krzywiecki,
X. Chen

Accountability

The controller shall be responsible for, and be able to demonstrate compliance with [the principles stated in GDPR]

Consequences

$\Rightarrow$ provable security and privacy

Reality

frequently, provable privacy has not been a design target
not even in research papers

Possible violations of GDPR principles

M. Kutyłowski, L. Krzywiecki, X. Chen

Malicious Prover

a Prover *A* may convince a third party *E* that an interaction between *A* and *B* has taken place

Malicious Verifier

a Verifier *B* may convince a third party *E* that a Prover *A* has authenticated itself against *B*,

Observer

a third party *E* may convince itself that an interaction between *A* and *B* has taken place
with no help from *A* and *B* but possibly with the help of the system provider, manufacturer of the hardware used by *A* and *B* etc.

Design target

Provable privacy goals

Protocol execution should not results in creating data that may help to violate privacy

Static Diffie-Hellman

Prover $\mathcal{V}$ shows that it holds the private key a corresponding to the public key $A = g^a$:

$\mathcal{V}$: chooses x at random , computes $X := g^x$, and sends X to the Prover $\mathcal{P}$.

$\mathcal{P}$: computes $Z := \mathcal{H}(X^a)$.

$\mathcal{P}$: sends Z to the Verifier $\mathcal{V}$.

$\mathcal{V}$: accepts iff $Z = \mathcal{H}(A^x)$.

Simulatability

$\mathcal{V}$ can create the answer of $\mathcal{P}$ by himself

so $\mathcal{V}$ cannot convince Eve that it has interacted with $\mathcal{P}$

unfortunately it is wrong!

DH Oracle

$\mathcal{V}$ may run the protocol as a CDH oracle

Convincing Eve about an interaction

1. Eve chooses x at random, computes $X := g^x$, $h := \mathcal{H}(t, \mathcal{H}(A^x))$, $C := \mathrm{Enc}_h(x)$ and sends (X, C, t) to $\mathcal{V}$
2. once $\mathcal{V}$ meets $\mathcal{P}$, then it sends the challenge X
3. on return of $\mathcal{H}(A^x)$ the Verifier $\mathcal{V}$ recomputes h, decrypts C to x'. If $X = g^{x'}$, then $\mathcal{V}$ accepts $\mathcal{P}$.
4. $\mathcal{V}$ sends x' to Eve as a proof of interaction with $\mathcal{P}$

Stinson-Wu protocol

M. Kutyłowski, L. Krzywiecki, X. Chen

$\mathcal{V}$ shows that it holds the private key a corresponding to $A = g^a$:

1. $\mathcal{V}$: chooses x at random, computes $X := g^x$, $Y := \mathcal{H}(A^x)$ and sends (X, Y) to the Prover $\mathcal{P}$.
2. $\mathcal{P}$: computes $Z := X^a$ and aborts if $Y \neq \mathcal{H}(Z)$.
3. $\mathcal{P}$: sends Z to the Verifier $\mathcal{V}$.
4. $\mathcal{V}$: accepts iff $Z = A^x$.

no CDH oracle

the innovation is that the Prover can see whether the discrete logarithm of X is known

Problem

- it does not say who knows the discrete logarithm of X.
- again, it might be Eve and not $\mathcal{V}$ $\Rightarrow$ a similar attack applies

Next step simulatability
work in progress

M. Kutyłowski, L. Krzywiecki, X. Chen

How to secure against dishonest Verifier/Prover?

- **a transcript of protocol execution should provide no proof that the Prover has been involved**
- this concerns not only regular executions but also executions with failures, with rogue challenges sent by the Verifier, etc.

Next-step simulatability

at any step of protocol execution, the Verifier can create the answer of the Prover himself

- regardless whether he follows the protocol specification,
- this concerns also aborting the protocol by the Prover.

Protocol example

work in progress

Setup

G is a group of a prime order q such that DL assumption holds,
g is a fixed generator of G

Key generation for user j

private key: randomly chosen $a_j < q$

public key: $A_j = g^{a_j}$

Protocol example
work in progress

M. Kutyłowski, L. Krzywiecki, X. Chen

Identification

$\mathcal{P}$ holds private key a_j and public key A_j,
$\mathcal{V}$ holds private key $\mathrm{sk}_{\mathcal{V}}$ and public key $A_{\mathcal{V}}$

$\mathcal{V}$: chooses $r \in G$ at random and calculates

$$h := g^{\mathcal{H}(r)} \cdot r, \quad w_j := A_j^{\mathcal{H}(r)}, \quad w_{\mathcal{V}} := A_{\mathcal{V}}^{\mathcal{H}(r)} .$$

$\mathcal{V}$: sends $(h, w_j, w_{\mathcal{V}})$ to $\mathcal{P}$.

$\mathcal{P}$: calculates $r' := h \cdot (w_j)^{-1/a_j \bmod q}$ and $z := \mathcal{H}(r')$.

$\mathcal{P}$: aborts if

$$h \neq g^z \cdot r' \quad \text{or} \quad w_j \neq A_j^z \quad \text{or} \quad w_{\mathcal{V}} \neq A_{\mathcal{V}}^z .$$

$\mathcal{P}$: computes $\rho := \mathcal{H}'(r')$ and sends ρ to $\mathcal{V}$.

$\mathcal{V}$: accepts iff $\rho = \mathcal{H}'(r)$.

the Prover knows that the Verifier can derive r' using $a_{\mathcal{V}}$ instead of a_j

Protocol example
work in progress

M. Kutyłowski, L. Krzywiecki, X. Chen

full version of the protocol

- the Prover must check that its interlocutor is the Verifier
- a kind of left-or-right game
- ... need to be careful to preserve the next-step simulatability

Conclusions

it is possible to defend the privacy threats

the protocol is still simple enough to meet practical limitations

Thanks for your attention!

Contact data

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