Motivation

use case: signed auctions

Can an auctioneer check the authenticity of signed bids using a resource-limited device? **YES**
**What is Server-Aided Verification?**

**Signature Scheme:**

- `SetUp(1^\lambda) \rightarrow gp`
- `KeyGen(gp) \rightarrow (pk, sk)`
- `Sign(sk, m) \rightarrow \sigma`

**Server-Aided Verification (SAV)**

**Example: the BLS [BonehLS04]**

- `gp = \text{BilinGroup}`
- `pk = g^{sk}, \quad sk \leftarrow \mathbb{Z}_p`
- `\sigma = \text{Hash}(m)^{sk}`
- `e(\sigma, g) \overset{?}{=} e(\text{Hash}(m), pk)`

**cost(Verify) > cost(Sign)**

---

Our Contributions

**use case: signed auctions**

- **bidder**
  - (signer)

- **cloud**
  - (server)

- **auctioneer**
  - (verifier)

1. **First model for single-round SAV**
   - Extend notion of unforgeability
     - (10€, blue_hippo)
     - (15€, grey_hyippo)

2. **First generic compiler for SAV**

3. **Introduce signer anonymity**

4. **Security results on our compiler for SAV**

New SAV for [BLS04], [Wat05] and the first SAV for [CLO4]
Single-Round SAV
Anonymous Single-Round SAV

Single-Round Server-Aided Verification (SAV) [this paper]

SAV.Init(1^λ) \rightarrow gp \quad SAV.KeyGen(gp) \rightarrow (pk, sk) \quad SAV.Vsetup(gp) \rightarrow (pb, pr)

SAV.ProbGen(pr, pk, m, σ) \rightarrow (ω, τ)

SAV.Verify(pr, pk, m, σ, ρ, τ)

SAV.Comp(pb, ω) \rightarrow ρ

Δ \in \{0, 1, \bot\}

Δ = \text{Verify}(pk, m, σ) \lor \bot

\text{Completeness}

\text{Efficiency}

\text{cost}(SAV.ProbGen) + \text{cost}(SAV.Verify) < \text{cost}(Verify)
The first compiler for SAV
Given a signature scheme $\Sigma$

1 - split Verify into two algorithms:

- $\text{Verify}_H$ (computationally heavy)
- $\text{Verify}_L$ (computationally light)

2 - off-load to the server $\text{Verify}_H$ using Verifiable Computation $\Gamma$

if $\Gamma$ returns 0, output $\Delta = \bot$

3 - check correctness of server’s output

4 - run $\text{Verify}_L$ using server’s output

if $\Gamma$ returns 1, output $\Delta \in \{0,1\}$

the outcome of $\text{Verify}_L$
Security Model for SAV
Anonymity in SAV: motivation

use case: signed auctions

 bidder

 Mrs Grey is bidding higher than Mr Blue .... interesting!!

 auctioneer

 (signer)

 (server)

 cloud

 SAV

 (10€, blue_hippo)

 (15€, grey_hippo)
Anonymity in SAV: motivation

**use case: signed auctions**

bidder

(auctioneer)

(signer)

cloud

(10€, blue_hippo)

(15€, grey_hippo)

Who is bidding higher?

.... *humpf*!!
Anonymity in SAV: definition (intuition)

use case: signed auctions

challenger

bidder

auctioneer

\( b \leftarrow \{0, 1\} \)

\( \sigma \leftarrow \text{SAV}.\text{Sign}(sk_b, m^*) \)

\((\hat{\omega}, \tau) \leftarrow \text{SAV}.\text{ProbGen}(pr, pk_b, m^*, \sigma) \)

\((m, i \in \{0, 1\}) \)

\( m^* \)

\( \omega \)

\( \hat{\omega} \)

(adversary)

\( \text{gp; pb} \)

\( \text{pk}_0, \text{pk}_1 \)

\( b' \in \{0, 1\} \)
Unforgeability in SAV : EUF

use case: signed auctions

I'm Mrs Grey and my bid is 25€

the bid should be discarded

Existential Forgery [ChowAS13]

\((m^*, \sigma^*, \rho^*)\) is a forgery if there is no query phase II and it holds that:

\((m^*, \cdot) \notin L\) and \(1 \leftarrow \text{SAV.\text{Verify}}(\text{pr, pk, } m^*, \sigma^*, \rho^*, \hat{\tau})\)
Unforgeability in SAV: ExEUF

Use case: signed auctions

(\(m^*, \sigma^*\)) (25€, grey_hippo)

The signature is valid

\(0 = \Delta \leftarrow \text{SAV}\)

I don't want Mrs Grey to buy this item

But it should be \(\Delta = 1\) since the bid was valid

What if the adversary wants to keep one signer out of the bid?

Extended Existential Forgery

\((m^*, \sigma^*, \rho^*)\) is a forgery if:

\((m^*, \cdot) \notin L\) and \(1 \leftarrow \text{SAV}.\text{Verify}(pr, pk, m^*, \sigma^*, \rho^*, \hat{\tau})\)

Or

\((m^*, \sigma^*) \in L\) and \(0 \leftarrow \text{SAV}.\text{Verify}(pr, pk, m^*, \sigma^*, \rho^*, \hat{\tau})\)
Security results on our compiler (in a nutshell)
SAV$^\Gamma_\Sigma$: security results

**THEOREM 1**

\[ \Sigma \text{ EUF } / \text{(SUF)} \]
\[ \Gamma \text{ secure } \implies SAV^\Gamma_\Sigma \text{ ExEUF } / \text{(ExSUF)} \]

**THEOREM 2**

\[ \Sigma \text{ correct } \]
\[ \Gamma \text{ secure } \implies SAV^\Gamma_\Sigma \text{ sound against collusion} \]

**THEOREM 3**

\[ \Sigma \text{ correct } \]
\[ \Gamma \text{ private } \implies SAV^\Gamma_\Sigma \text{ computationally anonymous} \]

**THEOREM 4**

\[ \Sigma \text{ adaptable } \]
\[ \Gamma \text{ correct } \implies SAV^\Gamma_\Sigma \text{ unconditionally anonymous} \]
### Comparison with previous work

Table 1. Comparison among our SAV schemes and previous works: Protocol I (Figure 3 in [25]), Protocol II (Figure 5 in [25]), Protocol III (Figure 4 in [25]), SAV-ZSS [15] (depicted in Figure 1 in [25]).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>collusion</strong></td>
<td>no [8]</td>
<td>no (C.3)</td>
<td>yes (6.1)</td>
<td>no (C.3)</td>
<td>yes (6.2)</td>
<td>no (C.3)</td>
<td>yes (6.3)</td>
</tr>
<tr>
<td><strong>anonymity</strong></td>
<td>no (C.4)</td>
<td>no (C.4)</td>
<td>no (6.1)</td>
<td>yes (C.4)</td>
<td>yes (6.2)</td>
<td>no (C.4)</td>
<td>yes (6.3)</td>
</tr>
</tbody>
</table>

[6]: S. Canard, J. Devigne, and O. Sanders. Delegating a pairing can be both secure and efficient. ACNS, 2014.
Take Away(s)

efficiently, securely and privately

Can an auctioneer check the authenticity of signed bids using a resource-limited device? **YES**

There exists a way to make SAV for any signature.

SAV can be singer-anonymous.

SAV can be *very* unforgeable.

Best security achieved employing verifiable computation.
Further directions

More than anonymity: signer indistinguishability?

Extend SAV to signature schemes other than pairing-based?

WANTED: more efficient VC schemes for bilinear pairing.

\[
\text{cost}(\text{SAV.ProbGen}) + \text{cost}(\text{SAV.Verify}) < \text{cost}(\text{Verify})
\]

\[
\text{cost}(\text{ProbGen}^\Gamma) + \text{cost}(\text{Verify}^\Gamma) < \text{cost}(\text{Verify}_H)
\]
Thank you for your attention!

Questions?

use case: signed auctions

Extend unforgeability notion for SAV

(10€, blue_hyppo)

(15€, grey_hyppo)

Introduce signer anonymity

First model for single-round SAV

First generic compiler for SAV

Security results on our compiler for SAV

New SAV for BLS04, Wat05 and the first SAV for CLO4

Elena Pagnin: www.cse.chalmers.se/~elenap