ARIEL UNIVERSITY

אוניברסיטת אריאל בשומרון

# Foundations of Modern Cryptography

Edward A. Hirsch

https://edwardahirsch.github.io/edwardahirsch

Neapolis University Pafos

Lecture 6: November 6, 2024

# Lecture Plan

- Digital signatures.

If the screen seems frozen and I do not respond,
please call me in Telegram.

# Task: Digital Signatures

Alice

public key, secret key

Bob

—

Charlie

# Task: Digital Signatures

Alice

public key, secret key

public key

Bob

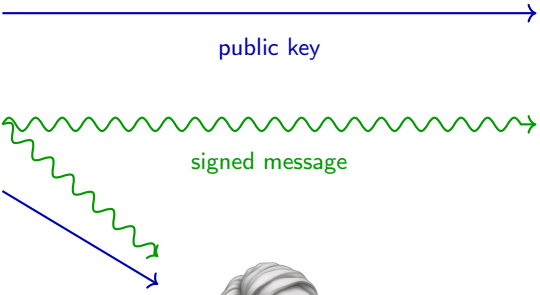
—

Charlie

# Task: Digital Signatures

Alice

public key, secret key




Bob

—

Charlie

# Task: Digital Signatures

Alice

public key, secret key

public key

signed message

fake message

Bob

—

Charlie

Charlie should be unable to sign (not to say recover the secret key) even though he has seen a lot of signed messages.

Pictures courtesy AI Emojis Generator

# Task: Digital Signatures

Alice

public key, secret key

public key

signed message

fake message

Bob

—

Charlie

Charlie should be unable to sign (not to say recover the secret key) even though he has seen a lot of signed messages.

Perhaps his partner even forced Alice to sign similar messages!

Pictures courtesy AI Emojis Generator

# Digital signatures

## Definition (Digital signatures scheme, DSS)

. . . consists of polynomial-time algorithms $G, S, V$:

- $G$: $(1^n, r_g) \mapsto (s, v)$ (secret signing key, public verification key),
- $S$: $(s, \texttt{msg}) \mapsto \texttt{sign}$ (signing algorithm),
- $V$: $(v, \texttt{msg}, \texttt{sign}) \mapsto$ yes or no,

Correctness:

- $\forall\ \texttt{msg}\ \mathbf{Pr}_{r_g}\{V_v(\texttt{msg}, S_s(\texttt{msg})) = 0\} = 0.$

# Digital signatures

## Definition (Digital signatures scheme, DSS)

. . . consists of polynomial-time algorithms $G, S, V$:

- $G\colon (1^n, r_g) \mapsto (s, v)$ (secret signing key, public verification key),
- $S\colon (s, \texttt{msg}) \mapsto \texttt{sign}$ (signing algorithm),
- $V\colon (v, \texttt{msg}, \texttt{sign}) \mapsto$ yes or no,

Correctness:

- $\forall\ \texttt{msg}\ \mathbf{Pr}_{r_g}\{V_v(\texttt{msg}, S_s(\texttt{msg})) = 0\} = 0.$

Security:

Recall: $A$ uses an oracle (black box)

- $\forall$ adversary $A^\bullet$, $\mathbf{Pr}\{A^{S_s}(v) = (\texttt{msg}, \texttt{sign}) : V_v(\texttt{msg}, \texttt{sign}) = 1\} < \varepsilon(n)$, where $A^{S_s}$ does not ask the oracle $S_s$ to sign this specific msg.

# Digital signatures

## Definition (Digital signatures scheme, DSS)

. . . consists of polynomial-time algorithms $G, S, V$:

- $G\colon (1^n, r_g) \mapsto (s, v)$ (secret signing key, public verification key),
- $S\colon (s, \texttt{msg}) \mapsto \texttt{sign}$ (signing algorithm),
- $V\colon (v, \texttt{msg}, \texttt{sign}) \mapsto$ yes or no,

Correctness:

- $\forall\ \texttt{msg}\ \mathbf{Pr}_{r_g}\{V_v(\texttt{msg}, S_s(\texttt{msg})) = 0\} = 0.$

Security:

Recall: $A$ uses an oracle (black box)

- $\forall$ adversary $A^\bullet$, $\mathbf{Pr}\{A^{S_s}(v) = (\texttt{msg}, \texttt{sign}) : V_v(\texttt{msg}, \texttt{sign}) = 1\} < \varepsilon(n)$, where $A^{S_s}$ does not ask the oracle $S_s$ to sign this specific msg.

- One-time scheme: a single-query adversary $A^{\bullet[1]}$.
- $\ell(n)$-bounded scheme: limited to messages of length $\ell(n)$.
- General schemes.

# Digital signatures

## Definition (Digital signatures scheme, DSS)

. . . consists of polynomial-time algorithms $G, S, V$:

- $G$: $(1^n, r_g) \mapsto (s, v)$ (secret signing key, public verification key),
- $S$: $(s, \texttt{msg}) \mapsto \texttt{sign}$ (signing algorithm),
- $V$: $(v, \texttt{msg}, \texttt{sign}) \mapsto$ yes or no,

Correctness:

- $\forall\ \texttt{msg}\ \mathbf{Pr}_{r_g}\{V_v(\texttt{msg}, S_s(\texttt{msg})) = 0\} = 0.$

Security:

Recall: $A$ uses an oracle (black box)

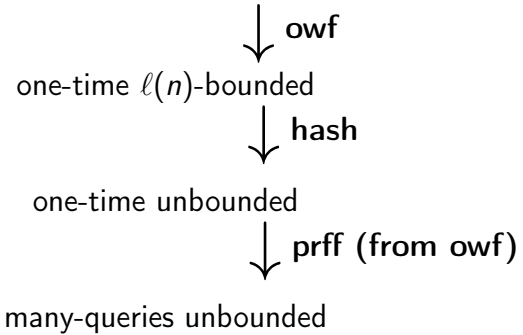
- $\forall$ adversary $A^\bullet$, $\mathbf{Pr}\{A^{S_s}(v) = (\texttt{msg}, \texttt{sign}) : V_v(\texttt{msg}, \texttt{sign}) = 1\} < \varepsilon(n)$, where $A^{S_s}$ does not ask the oracle $S_s$ to sign this specific $\texttt{msg}$.

- One-time scheme: a single-query adversary $A^{\bullet[1]}$.
- $\ell(n)$-bounded scheme: limited to messages of length $\ell(n)$.
- General schemes.

## Exercise

Does it matter whether $S, V$ are deterministic or randomized?

# Workflow


owf
one-time $\ell(n)$-bounded
hash
one-time unbounded
prff (from owf)
many-queries unbounded

# One-time bounded DSS: Construction

- Document: original document + randomness.

# One-time bounded DSS: Construction

- Document: original document + randomness.
- Public key `pk`: a string in the image of a one-way function.
- Signature: a pre-image under a one-way function.

# One-time bounded DSS: Construction

- Document: original document + randomness.
- Public key `pk`: a string in the image of a one-way function.
- Signature: a pre-image under a one-way function.
- Only we know a pre-image (we selected it before publishing `pk`).

# One-time bounded DSS: Construction

- Document: original document + randomness.
- Public key pk: a string in the image of a one-way function.
- Signature: a pre-image under a one-way function.
- Only we know a pre-image (we selected it before publishing pk).

## Theorem

Let $f$ be a owf, $m = \ell(n)$ is a polynomial. Let $s_k^i \leftarrow U_n$, compute $v_k^i = f(s_k^i)$,

$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, s_m^0, s_m^1)}_{\mathtt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, v_m^0, v_m^1)}_{\mathtt{pk}}).$$

# One-time bounded DSS: Construction

- Document: original document + randomness.
- Public key pk: a string in the image of a one-way function.
- Signature: a pre-image under a one-way function.
- Only we know a pre-image (we selected it before publishing pk).

## Theorem

Let $f$ be a owf, $m = \ell(n)$ is a polynomial. Let $s_k^i \leftarrow U_n$, compute $v_k^i = f(s_k^i)$,

$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, s_m^0, s_m^1)}_{\mathtt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, v_m^0, v_m^1)}_{\mathtt{pk}}).$$

Let

$$S_s(\underbrace{\beta_1 \ldots \beta_m}_{\mathtt{msg}}) = (s_1^{\beta_1}, \ldots, s_m^{\beta_m}).$$

Let $V(\mathtt{msg}, \mathtt{sign})$ check that

$$\forall i \quad f(\mathtt{sign}_i) = v_i^{\beta_i}.$$

Then $(G, S, V)$ is a secure one-time $\ell(n)$-DSS.

# One-time bounded DSS: Proof

- Let us invert $f$ using the adversary $A^\bullet$ for our DSS.
- Given $y$, find $f^{-1}(y)$.

# One-time bounded DSS: Proof

- Let us invert $f$ using the adversary $A^\bullet$ for our DSS.
- Given $y$, find $f^{-1}(y)$.
- Pick random $k, b$. Generate

$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, \cancel{s_k^b}, \ldots, s_m^0, s_m^1)}_{\texttt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk}}).$$

# One-time bounded DSS: Proof

- Let us invert $f$ using the adversary $A^\bullet$ for our DSS.
- Given $y$, find $f^{-1}(y)$.
- Pick random $k, b$. Generate

$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, \cancel{s_k^b}, \ldots, s_m^0, s_m^1)}_{\texttt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk}}).$$

- Not a correct key, but who can notice?

# One-time bounded DSS: Proof

- Let us invert $f$ using the adversary $A^\bullet$ for our DSS.
- Given $y$, find $f^{-1}(y)$.
- Pick random $k, b$. Generate

$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, \cancel{s_k^b}, \ldots, s_m^0, s_m^1)}_{\texttt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk}}).$$

- Not a correct key, but who can notice?
- $A^\bullet$ asks as to sign a message $x$; if $\boxed{x_k \neq b}$, we can do that (prob. $\frac{1}{2}$ ).

# One-time bounded DSS: Proof

- Let us invert $f$ using the adversary $A^\bullet$ for our DSS.
- Given $y$, find $f^{-1}(y)$.
- Pick random $k, b$. Generate

$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, \cancel{s_k^b}, \ldots, s_m^0, s_m^1)}_{\texttt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk}}).$$

- Not a correct key, but who can notice?
- $A^\bullet$ asks as to sign a message $x$; if $\boxed{x_k \neq b}$, we can do that (prob. $\frac{1}{2}$ ???).
- What if $A$ can guess $k, b$?

# One-time bounded DSS: Proof

- Let us invert $f$ using the adversary $A^\bullet$ for our DSS.
- Given $y$, find $f^{-1}(y)$.
- Pick random $k, b$. Generate

$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, \cancel{s_k^b}, \ldots, s_m^0, s_m^1)}_{\texttt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk}}).$$

- Not a correct key, but who can notice?
- $A^\bullet$ asks as to sign a message $x$; if $\boxed{x_k \neq b}$, we can do that (prob. $\frac{1}{2}$! ).
- What if $A$ can guess $k, b$?
- $A$ sees:

$$\underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk?}},$$

$y \leftarrow f(U)$, no one can distinguish from pk from $G$, the same distribution.

# One-time bounded DSS: Proof

- Let us invert $f$ using the adversary $A^\bullet$ for our DSS.
- Given $y$, find $f^{-1}(y)$.
- Pick random $k, b$. Generate

$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, \cancel{s_k^b}, \ldots, s_m^0, s_m^1)}_{\texttt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk}}).$$

- Not a correct key, but who can notice?
- $A^\bullet$ asks as to sign a message $x$; if $\boxed{x_k \neq b}$, we can do that (prob. $\frac{1}{2}$ ).
- What if $A$ can guess $k, b$?
- $A$ sees:

$$\underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk?}},$$

  $y \leftarrow f(U)$, no one can distinguish from pk from $G$, the same distribution.
- Do we get $f^{-1}(y)$? $A$'s forged message $z \neq x$ differs...

# One-time bounded DSS: Proof

- Let us invert $f$ using the adversary $A^\bullet$ for our DSS.
- Given $y$, find $f^{-1}(y)$.
- Pick random $k, b$. Generate

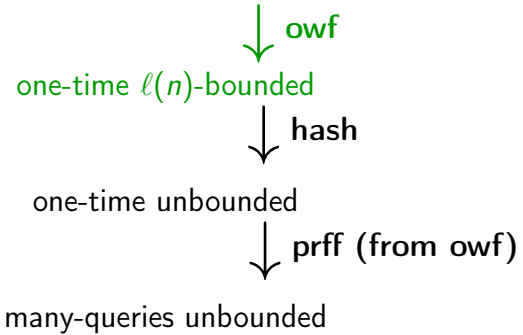
$$G(1^n) = (\underbrace{(s_1^0, s_1^1, \ldots, \cancel{s_k^b}, \ldots, s_m^0, s_m^1)}_{\texttt{sk}}, \underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk}}).$$

- Not a correct key, but who can notice?
- $A^\bullet$ asks as to sign a message $x$; if $\boxed{x_k \neq b}$, we can do that (prob. $\frac{1}{2}$ ).
- What if $A$ can guess $k, b$?
- $A$ sees:

$$\underbrace{(v_1^0, v_1^1, \ldots, y, \ldots, v_m^0, v_m^1)}_{\texttt{pk?}},$$

  $y \leftarrow f(U)$, no one can distinguish from pk from $G$, the same distribution.
- Do we get $f^{-1}(y)$? $A$'s forged message $z \neq x$ differs... in the $k$-th bit $\boxed{z_k \neq x_k}$ with prob. $\frac{1}{m}$.

# Workflow


owf
one-time $\ell(n)$-bounded
hash
one-time unbounded
prff (from owf)
many-queries unbounded

# One-time **un**bounded scheme: Idea

## Hash-and-sign paradigm; collision-resistant hash-function family

- We can sign $\ell(n)$ bits.
- We need to sign many bits.

# One-time **un**bounded scheme: Idea

## Hash-and-sign paradigm; collision-resistant hash-function family

- We can sign $\ell(n)$ bits.
- We need to sign many bits.
- Solution: sign a $\ell(n)$-bit hash.

# One-time **un**bounded scheme: Idea

## Hash-and-sign paradigm; collision-resistant hash-function family

- We can sign $\ell(n)$ bits.
- We need to sign many bits.
- Solution: sign a $\ell(n)$-bit hash.
- It works if the hashes are different,

# One-time **un**bounded scheme: Idea

## Hash-and-sign paradigm; collision-resistant hash-function family

- We can sign $\ell(n)$ bits.
- We need to sign many bits.
- Solution: sign a $\ell(n)$-bit hash.
- It works if the hashes are different,
- ... which is not possible.

# One-time **un**bounded scheme: Idea

Hash-and-sign paradigm; collision-resistant hash-function family

- We can sign $\ell(n)$ bits.
- We need to sign many bits.
- Solution: sign a $\ell(n)$-bit hash.
- It works if the hashes are different,
- ... which is not possible.
- It works if identical hashes are difficult to find:

## Definition (Collision-resistant hash-function family, crhff)

$\ell(m)$-size crhff $\{h_\xi\}_\xi$ is given by polynomial-time $Z, H$ such that

- $Z\colon (1^n, r_Z) \mapsto \xi$, where this "index" $\xi$ is not too short ($\xi \in \{0,1\}^{s(n)}$, $n = \mathrm{poly}(s(n))$)

# One-time **un**bounded scheme: Idea

Hash-and-sign paradigm; collision-resistant hash-function family

- We can sign $\ell(n)$ bits.
- We need to sign many bits.
- Solution: sign a $\ell(n)$-bit hash.
- It works if the hashes are different,
- ... which is not possible.
- It works if identical hashes are difficult to find:

## Definition (Collision-resistant hash-function family, crhff)

$\ell(m)$-size crhff $\{h_\xi\}_\xi$ is given by polynomial-time $Z, H$ such that

- $Z\colon (1^n, r_Z) \mapsto \xi$, where this "index" $\xi$ is not too short ($\xi \in \{0,1\}^{s(n)}$, $n = \mathrm{poly}(s(n))$)
- $h_\xi\colon \{0,1\}^* \to \{0,1\}^{\ell(|\xi|)}$
- $H(\xi, x) = h_\xi(x)$

# One-time **un**bounded scheme: Idea

Hash-and-sign paradigm; collision-resistant hash-function family

- We can sign $\ell(n)$ bits.
- We need to sign many bits.
- Solution: sign a $\ell(n)$-bit hash.
- It works if the hashes are different,
- ... which is not possible.
- It works if identical hashes are difficult to find:

## Definition (Collision-resistant hash-function family, crhff)

$\ell(m)$-size crhff $\{h_\xi\}_\xi$ is given by polynomial-time $Z, H$ such that

- $Z\colon (1^n, r_Z) \mapsto \xi$, where this "index" $\xi$ is not too short ($\xi \in \{0,1\}^{s(n)}$, $n = \mathrm{poly}(s(n))$)
- $h_\xi\colon \{0,1\}^* \to \{0,1\}^{\ell(|\xi|)}$
- $H(\xi, x) = h_\xi(x)$
- $\forall$ adversary $A$ $\quad \mathbf{Pr}\{A(\xi) = (y, y') : h_\xi(y) = h_\xi(y'),\ y \neq y'\} < \varepsilon(n)$

## Exercise

1. Give an equivalent definition without $H$.
2. Show that crhff can be constructed from owf.

# One-time **un**bounded scheme: Construction

## Theorem

Assume that $(G, S, V)$ is a secure one-time $\ell(m)$-bounded DSS and $\{h_\xi\}_\xi$ is a $\ell(m)$-crhff generated by $Z$. Then the following construction provides a secure one-time unbounded DSS:

- Generate $(s, v)$ using $G$, generate $\xi$ using $Z$
- Signing key `sk`: $(\xi, s)$
- Verification key `pk`: $(\xi, v)$

# One-time **un**bounded scheme: Construction

## Theorem

Assume that $(G, S, V)$ is a secure one-time $\ell(m)$-bounded DSS and $\{h_\xi\}_\xi$ is a $\ell(m)$-crhff generated by $Z$. Then the following construction provides a secure one-time unbounded DSS:

- Generate $(s, v)$ using $G$, generate $\xi$ using $Z$
- Signing key $\mathtt{sk}$: $(\xi, s)$
- Verification key $\mathtt{pk}$: $(\xi, v)$
- $S'_{(\xi,s)}(\mathtt{msg}) = S_s(h_\xi(\mathtt{msg}))$.

# One-time **un**bounded scheme: Construction

## Theorem

Assume that $(G, S, V)$ is a secure one-time $\ell(m)$-bounded DSS and $\{h_\xi\}_\xi$ is a $\ell(m)$-crhff generated by $Z$. Then the following construction provides a secure one-time unbounded DSS:

- Generate $(s, v)$ using $G$, generate $\xi$ using $Z$
- Signing key $\mathtt{sk}$: $(\xi, s)$
- Verification key $\mathtt{pk}$: $(\xi, v)$
- $S'_{(\xi,s)}(\mathtt{msg}) = S_s(h_\xi(\mathtt{msg}))$.
- $V'_{(\xi,v)}(\mathtt{msg}, \mathtt{sign}) = V_v(h_\xi(\mathtt{msg}), \mathtt{sign})$.

# One-time **un**bounded scheme: Construction

## Theorem

Assume that $(G, S, V)$ is a secure one-time $\ell(m)$-bounded DSS and $\{h_\xi\}_\xi$ is a $\ell(m)$-crhff generated by $Z$. Then the following construction provides a secure one-time unbounded DSS:

- Generate $(s, v)$ using $G$, generate $\xi$ using $Z$
- Signing key $\mathtt{sk}$: $(\xi, s)$
- Verification key $\mathtt{pk}$: $(\xi, v)$
- $S'_{(\xi,s)}(\mathtt{msg}) = S_s(h_\xi(\mathtt{msg}))$.
- $V'_{(\xi,v)}(\mathtt{msg}, \mathtt{sign}) = V_v(h_\xi(\mathtt{msg}), \mathtt{sign})$.

## Exercise

Prove it (hint: we must break *either* $(G, S, V)$ or $\{h_\xi\}_\xi$).

# One-time **un**bounded scheme: Idea, version 2

## Hash-and-sign paradigm, revisited; one-way hash-function family

- Include the hash function in the message (not the keys).
- $S'_s(\texttt{msg}) = (\xi,\ S_s(\xi \circ h_\xi(\texttt{msg})))$
  - $\xi$ is regenerated every time,

# One-time **un**bounded scheme: Idea, version 2

Hash-and-sign paradigm, revisited; one-way hash-function family

- Include the hash function in the message (not the keys).
- $S'_s(\texttt{msg}) = (\xi,\ S_s(\xi \circ h_\xi(\texttt{msg})))$
  - $\xi$ is regenerated every time,

## Definition (One-way hash-function family, owhff)

$\ell(m)$-size owhff $\{h_\xi\}_\xi$ is given by polynomial-time $Z, H$ such that

- $Z\colon (1^n, r_Z) \mapsto \xi$, where this "index" $\xi$ is not too short ($\xi \in \{0,1\}^{s(n)}$, $n = \mathrm{poly}(s(n))$)
- $h_\xi\colon \{0,1\}^* \to \{0,1\}^{\ell(|\xi|)}$
- $H(\xi, x) = h_\xi(x)$

# One-time **un**bounded scheme: Idea, version 2

Hash-and-sign paradigm, revisited; one-way hash-function family

- Include the hash function in the message (not the keys).
- $S'_s(\mathtt{msg}) = (\xi,\ S_s(\xi \circ h_\xi(\mathtt{msg})))$
  - $\xi$ is regenerated every time,

## Definition (One-way hash-function family, owhff)

$\ell(m)$-size owhff $\{h_\xi\}_\xi$ is given by polynomial-time $Z, H$ such that

- $Z\colon (1^n, r_Z) \mapsto \xi$, where this "index" $\xi$ is not too short ($\xi \in \{0,1\}^{s(n)}$, $n = \mathrm{poly}(s(n))$)
- $h_\xi\colon \{0,1\}^* \to \{0,1\}^{\ell(|\xi|)}$
- $H(\xi, x) = h_\xi(x)$
- $\forall$ adversaries $M, A$

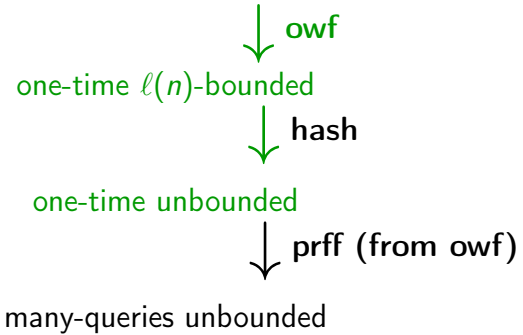
$$\mathbf{Pr}_{y \leftarrow M(1^n),\ \xi \leftarrow Z(1^n)}\{A(\xi) = y' : h_\xi(y) = h_\xi(y'),\ y \neq y'\} < \varepsilon(n)$$

- $y$ is chosen before $\xi$.

## Exercise

1. Give an equivalent definition without $H$.
2. Show that owhff can be constructed from owf.
3. Prove that it is enough for securing the DSS construction.

# Workflow


owf
one-time $\ell(n)$-bounded
hash
one-time unbounded
prff (from owf)
many-queries unbounded

# Many-time DSS: Idea

## From one-time DSS

- Idea: new public key is signed by old public key.

# Many-time DSS: Idea

## From one-time DSS

- Idea: new public key is signed by old public key.
- Better idea: sign two new keys $v_0, v_1 \Rightarrow$ potentially $2^n$ new keys.
- Binary tree of keys!

# Many-time DSS: Idea

## From one-time DSS

- Idea: new public key is signed by old public key.
- Better idea: sign two new keys $v_0, v_1 \Rightarrow$ potentially $2^n$ new keys.
- Binary tree of keys!
- No memory $\Rightarrow$ will regenerate key like $v_{0100010}$.

# Many-time DSS: Idea

## From one-time DSS

- Idea: new public key is signed by old public key.
- Better idea: sign two new keys $v_0, v_1 \Rightarrow$ potentially $2^n$ new keys.
- Binary tree of keys!
- No memory $\Rightarrow$ will regenerate key like $v_{0100010}$.
- To avoid signing two messages by the same key use prff instead of a truly random path. (Otherwise add this negligible probability of forgery.)

# Many-time DSS: Idea

## From one-time DSS

- Idea: new public key is signed by old public key.
- Better idea: sign two new keys $v_0, v_1 \Rightarrow$ potentially $2^n$ new keys.
- Binary tree of keys!
- No memory $\Rightarrow$ will regenerate key like $v_{0100010}$.
- To avoid signing two messages by the same key use prff instead of a truly random path. (Otherwise add this negligible probability of forgery.)
- We will still be using prff instead of the true randomness for $G$ and $S$ to be able to regenerate keys (and their certificates) without memory.

# Many-time DSS: Construction

## From one-time DSS and pseudorandom function family

- Use one-time $(G, S, V)$ (w.l.o.g., $G$ and $S$ require at most $n + 3$ random bits) and prff $Z$:
  - $(s, v) \leftarrow G$, "root key"
  - $\zeta \leftarrow Z$

# Many-time DSS: Construction

## From one-time DSS and pseudorandom function family

- Use one-time $(G, S, V)$ (w.l.o.g., $G$ and $S$ require at most $n + 3$ random bits) and prff $Z$:
  - $(s, v) \leftarrow G$, "root key"
  - $\zeta \leftarrow Z$
- Signing key `sk`: $(\zeta, s)$
- Verification key `pk`: $v$

# Many-time DSS: Construction

From one-time DSS and pseudorandom function family

- Use one-time $(G, S, V)$ (w.l.o.g., $G$ and $S$ require at most $n+3$ random bits) and prff $Z$:
  - $(s, v) \leftarrow G$, "root key"
  - $\zeta \leftarrow Z$
- Signing key $\mathtt{sk}$: $(\zeta, s)$
- Verification key $\mathtt{pk}$: $v$
- Signing algorithm
  - Select path $\sigma = \sigma_1 \dots \sigma_n \in \{0,1\}^n$ from the root key to a leaf either at random or using $\zeta(110\xi(\mathtt{msg}))$.
  - Generate all the keys (both left and right) for prefixes of this path ($i = 0, 1, \dots, n-1$):

$$(s_{\sigma'0}, v_{\sigma'0}) \leftarrow G(1^n, \zeta(10^{n-i+1}\sigma'0)) \text{ for } \sigma' = \sigma_1 \dots \sigma_i,$$
$$(s_{\sigma'1}, v_{\sigma'1}) \leftarrow G(1^n, \zeta(10^{n-i+1}\sigma'1)) \text{ for } \sigma' = \sigma_1 \dots \sigma_i,$$

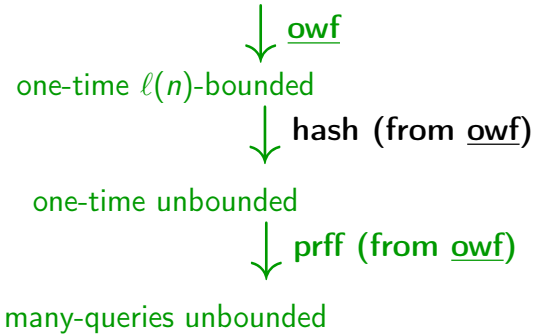
  - Given keys $(s_\sigma, v_\sigma)$:

$$S_{s_\sigma}(\mathtt{msg}, \underbrace{\zeta(00^{\cdots}\sigma)}_{\text{randomness for } S}).$$

  - Keys are signed sequentially:

$$\begin{array}{llll} (v_0, & v_1, & S_s(v_0 & \circ v_1 \quad , \zeta(00^{n+2}))), \\ (v_{\sigma_1 0}, & v_{\sigma_1 1}, & S_{s_{\sigma_1}}(v_{\sigma_1 0} & \circ v_{\sigma_1 1} \quad , \zeta(00^{n+1}\sigma_1))), \\ & \dots & & \\ (v_{\sigma_1 \dots \sigma_{n-1} 0}, & v_{\sigma_1 \dots \sigma_{n-1} 1}, & S_{s_{\sigma_1 \dots \sigma_{n-1}}}(v_{\sigma_1 \dots \sigma_{n-1} 0} & \circ v_{\sigma_1 \dots \sigma_{n-1} 1} \quad , \zeta(00^3\sigma_1 \dots \sigma_{n-1}))). \end{array}$$

# Workflow


owf
one-time $\ell(n)$-bounded
hash (from owf)
one-time unbounded
prff (from owf)
many-queries unbounded

# Reminder: Is Cryptography Possible?

## Impagliazzo's Worlds

Russell Impagliazzo

(taken at CSR-2006, SPb, by A. Kulikov)

Algorithmica

**P** = **NP**, no crypto

**P** ≠ **NP**, how to use?

Heuristica

Easy on the average

Trapdoor functions

Cryptomania

public-key encryption

digital money

Hard on the average

. . . also hard to generate

Pessiland

No one-way functions

One-way functions

Minicrypt

digital signatures

private-key crypto

# Takeout

- One-time length-bounded digital signatures from owf.
- One-time unbounded digital signatures, two constructions:
  - collision-resistant hash-function family,
  - one-way hash-function family.
- Multiple-query unbounded digital signatures (using prff).

# Takeout

- One-time length-bounded digital signatures from owf.
- One-time unbounded digital signatures, two constructions:
  - collision-resistant hash-function family,
  - one-way hash-function family.
- Multiple-query unbounded digital signatures (using prff).

*Coming next:*

- *DEADLINE (homework).*
- *Midterm.*