Signature schemes

Definition of signatures schemes and of CMA security The Schnorr signature scheme, you can do many details here, such as the proof that you cannot cheat the underlying interactive game with better than 1/q probability, and the full story on how you derive the signature scheme from the interactive game. Or you can just do the spec of the scheme, giving you time for something else, such as RSA+hash signatures and the proof that secure hash + secure signature scheme is secure. Or you can do the one-time signatures based on hash functions and the proof that they are secure.

The "Elevator Pitch" (Synthesis)

> **Explain Digital Signature Schemes to a peer in 3 sentences. Focus on WHY we use it, not just how.**

Core Vocabulary & Syntax (Total Recall)

Digital Signature System (G, S, V)

> A *Digital Signature System* (G, S, V) is defined, by
>
> - A probabilistic key generation algorithm G which gets a security parameter k as input and produces as output a pair of keys (pk, sk), the public and the secret key.
> - Algorithm S gets input a message m and the secret key sk and produces a signature $S_{sk}(m)$.
> - Finally algorithm V gets as input a signature s, a message m and public key pk, and outputs $V_{pk}(s, m)$ which is equal to *accept* or *reject*.
>
> It is required that we always have $V_{pk}(S_{sk}(m), m) = accept$.

CMA-security of DSSs

> Let's define CMA security by an oracle game (very similarly to MACs).

> The oracle runs G on input k to get sk, pk.
> We give pk to the adversary E, and keep sk inside the oracle.
> The adversary can submit as many messages as he wants and for each message m he will get $S_{sk}(m)$ back.
>
> The adversary wins the game if he produces m_0, s_0, such that m_0 is not one of the messages the oracle was asked to sign, and that $V_{pk}(s_0, m_0) = accept$.
>
> The probability that E wins is a function of the security parameter k and is called $Adv_E(k)$.

> **Definition 12.1 (Security for Signature Schemes)** We say that a signature scheme is CMA-secure if for any probabilistic polynomial time adversary E, if $Adv_E(k)$ is negligible as a function of k. This is the strongest sense in which a signature system can be secure.

Schnorr signature scheme

> The Schnorr signature scheme works in a subgroup of $\mathbb{Z}_p^*$
>
> **Parameters**:
>
> - Primes p, q where q divides $p-1$ (i.e., $q|(p-1)$)
> - Element $\alpha \in \mathbb{Z}_p^*$ that is q bits long ($|\alpha| = q$)
>
> > **Getting** α: Start with generator α_0 of $\mathbb{Z}_p^*$, then compute $\alpha = \alpha_0^{(p-1)/q} \mod p$. This gives an element of order exactly q.
>
> So the computation happens in $\mathbb{Z}_p$ (providing security), but we only need to transmit/store values from $\mathbb{Z}_q$ (providing efficiency).

Interactive game (Schnorr)

> **Purpose**: A game where the *signer* proves to a *verifier* that they know the secret key s — without revealing what s is. This is an example of a *zero-knowledge proof (of knowledge)*.

> **The Protocol** (given public key p, q, α, β):
>
> 1. **Signer** $\rightarrow$ **Verifier**: Send $c = \alpha^r$ where $r \in \mathbb{Z}_q$ is random
> 2. **Verifier** $\rightarrow$ **Signer**: Send random challenge $e \in \mathbb{Z}_q$
> 3. **Signer** $\rightarrow$ **Verifier**: Send response $z = (r + es) \mod q$
> 4. **Verifier checks**: $\alpha^z \overset{?}{=} c\beta^e \mod p$
>
> If the check passes, the verifier accepts (believes the signer knows s).

> **Correctness**: If the signer follows the protocol honestly, the verifier always accepts:
>
> $$c\beta^e = \alpha^r(\alpha^s)^e = \alpha^{r+es} = \alpha^z \pmod p$$

> **Security**: If the signer doesn't know s, they can only make the verifier accept with probability $1/q$ (negligible). The verifier only sees $z = (r + es) \mod q$, where the random r masks the secret part es.
>
> **Proof hint**: If a cheating signer can correctly answer *two different* challenges $e \neq e'$, then by dividing the two verification equations we get $\alpha^{z-z'} = \beta^{e-e'}$, which allows us to compute $s = (z - z')(e - e')^{-1} \mod q$. So any successful cheater who can answer more than one challenge actually *knows* s, meaning a cheater without knowledge can only succeed by correctly guessing which single challenge will be sent.

Signature scheme

> The game above is nice and all, but it is not desired that the signer always has to be reachable.

> **To make it non-interactive**:
> Replace the verifier's random challenge $e$ with $e = h(c, m)$ using a hash function $h : \{0,1\}^* \to \mathbb{Z}_q$.
> This links the signature to message m, and in the random oracle model, $h(c, m)$ is effectively random (like the interactive game).

> **The Scheme**:
>
> - **Generate Keys**: $pk = (h, p, q, \alpha, \beta = \alpha^s)$, $sk = s$ (where $s \in \mathbb{Z}_q$ random)
> - **Sign**(m): Pick random $r \in \mathbb{Z}_q$, compute $c = \alpha^r$, output (e, z) where $e = h(c, m)$ and $z = (r + es) \bmod q$
> - **Verify**$(m, (e, z))$: Compute $c = \alpha^z \beta^{-e}$, check $e \stackrel{?}{=} h(c, m)$

> **Correctness**: $\alpha^z \beta^{-e} = \alpha^{r+es} \cdot \alpha^{-es} = \alpha^r = c$, so $h(c, m) = e$

> **Security**: In the random oracle model, if Schnorr is NOT CMA-secure, then we can compute discrete logarithms.
>
> **Contrapositive**: If DL is hard in the subgroup generated by α, then Schnorr is CMA-secure (assuming ROM for $h$).
>
> **Intuition**: Seeing genuine signatures doesn't help an adversary — each signature is like an execution of the interactive game, revealing nothing about s. To forge a signature, the adversary would need to win the interactive game without knowing s, which requires computing discrete logs.

Logic Flow / Mechanism (Process)

Schnorr Interactive Game

Derive the steps of the interactive game that proves knowledge of the secret key$a$without revealing it. Provide the verifier's check equation.

1. Signer sends$c = \alpha^r$to verifier, where$r \in \mathbb{Z}_q$random.
2. [____________________]
3. [____________________]
4. Verifier checks$\alpha^z \stackrel{?}{=} c \beta^e \mod p$.

From Interactive Game to Schnorr Signature Scheme

Fill in the transformation steps using hash function$h$.

1. Replace verifier's random$e$with$e = h(c, m)$.
2. [____________________]
3. [____________________]

4. Verification: Compute$c = \alpha^z \beta^{-e} \mod p$, check$e \stackrel{?}{=} h(c, m)$.

Full Domain Hash Security Proof (High-Level)

Outline the contradiction proof structure for Theorem 12.2.

1. Assume PPT adversary$A$breaks CMA security with probability$\epsilon$.
2. [____________________]
3. [____________________] 4.*B*'s success probability$\approx \frac{1}{e} \frac{\epsilon}{q_{sig} + 1}$, non-negligible.

The "Exam Trap" (Distinctions)

Distinguish between plain RSA signatures and RSA + Full Domain Hash based on: (1) CMA security vulnerability, (2) attack method, (3) role of hash function.

Criterion	Plain RSA	RSA + Full Domain Hash
CMA Attack	[Student: Describe existential forgery]	[Student: Why attack fails]
Hash Role	[Student: None]	[Student: Collision intractability implication]
Proof Model	[Student: Insecure]	[Student: Random oracle + RSA assumption]

Exam Simulation

Oral Exam Question 1: "Walk me through the Schnorr interactive game. Prove why a cheater succeeds with probability at most$1/q$. Derive the secret key$a$if the cheater answers two different challenges correctly."

Oral Exam Question 2: "Present the Full Domain Hash scheme. Give the security proof intuition under random oracle model and RSA assumption. Compute$B$'s optimal success probability parameter$p$."

Oral Exam Question 3: "Explain Theorem 12.4: If$H$is collision intractable and$\Sigma$is secure, then combined$\Sigma'$is secure. Walk through the reduction: given breaker$E'*of*\Sigma'$, how does$E$either forge$\Sigma$or find hash collision?"

Source Map

- CryptographyV6.pdf | Page 147 | Covers: Definition of Digital Signature Schemes
- CryptographyV6.pdf | Page 148 | Covers: CMA security definition
- CryptographyV6.pdf | Page 149-151 | Covers: RSA + Full Domain Hash scheme and security proof (Theorem 12.2)

- CryptographyV6.pdf | Page 152-154 | Covers: Schnorr interactive game, derivation to signature scheme, security lemma
- CryptographyV6.pdf | Page 156-157 | Covers: General hash + signature combination (Theorem 12.4)

Source data missing for: One-time signatures from hash functions and their security proof.