Recall: A digital signature scheme \( w \), msg space \( M \) consists of ppt alg: \( (\text{KeyGen}, \text{Sign}, \text{Verify}) \)

- \( \text{KeyGen}(1^n) \) generates \( (PK, SK) \)
- \( \text{Sign}(SK, m) \) generates a signature \( \sigma \)
- \( \text{Verify}(PK, m, \sigma) = 0/1 \) ("acc" or "rej")

Correctness: \[ \forall m \in M \text{ for } (PK, SK) \leftarrow \text{KeyGen}(1^n) \]
\[ \Pr[\text{Verify}(PK, m, \text{Sign}(SK, m)) = 1] = 1 \]

Security (against adaptive chosen msg attacks):
\[ \forall \text{ ppt Adv, given } PK \text{ and oracle to } \text{Sign}(SK, \cdot), \text{ for } \]
\[ \forall \text{ ppt Adv, given } PK, SK \text{ and oracle to } \text{Sign}(SK, \cdot), \text{ denoting by } m_1, \ldots, m_k \text{ its oracle queries, the prob that } A \text{ outputs } (m^*, \sigma^*) \text{ s.t. } m^* \neq m_i \text{ & Verify}(PK, m^*, \sigma^*) = 1 \text{ is negl.} \]

Last lecture: RSA digital sig scheme
Follows Diffie-Hellman blue print:
\[ f_{n,e} : \mathbb{Z}_n \to \mathbb{Z}_n \quad x \mapsto x^e \mod n \]
\[ \text{Sign}(SK, m) = f^{-1}_{n,e}(m) = m^d \mod n \]
\( (n, d) \) e.d = 1 mod(e(n)).
Correctness: $\forall m \in \mathbb{Z}_n. (m^d)^e = m^{de} = m \mod n \checkmark$

Not secure: Given $\text{Sign}(sk, m) = m^d \mod n$

one can easily sign $m^2 \mod n \rightarrow (m^d)^2 \mod n$.

To make RSA secure use hash & sign:

**Hash & Sign**

Rather than signing $m$, sign $h(m)$, where $h$ is a hash function (part of the public key).

* Better efficiency: Hashing is extremely efficient compared to signing.

* Allow flexibility: Signing any msg in $\{0,1\}^*$.

* Interestingly: Useful for security.

Claim: If $(\text{KeyGen, Sign, Verify})$ is secure & $H = \{ h_k \}$ is a collision resistant hash family

then the hash & sign version of $(\text{KeyGen, Sign, Verify})$

is also secure.
Moreover: Hash & Sign paradigm enhances security for RSA

**Hash & Sign with RSA**

\[
\text{Sign (} (n,d,h), m) = h(m)^d \mod n
\]

\[
\text{Verify (} (n,e,h), m, \sigma) = 1 \iff 0^e = h(m) \mod n
\]

**Is this secure?** Depends on \( h \). \text{[not secure if } \text{in } h \text{ is not easy}\]

**It is secure in the Random Oracle Model** \text{[Bellare-Rogaway 93]}

\text{aka. Full Domain Hash (FDH)}

**Intuition:** Pairs \((m_i, r_i)\) are dist. like \((m_i, r_i^{\text{random}})\) where

\[
h(m_i) = r_i^e \mod n
\]

Doesn't give any useful info in ROH \(\equiv\) Can be simulated

If \( \text{Adv} \) generates \((m^*, \sigma^*)\) \(\Rightarrow\) \( \text{Adv} \) breaks RSA \((\text{in ROH})\)
Security reduction is not tight...

Loosely speaking, if RSA function is \((t', \varepsilon')\)-secure (i.e., \(\text{Adv}\) running in time \(t'\) can invert w.p. \(\leq \varepsilon'\))

then FDH scheme is \((t, g_{\text{sig}}, g_{\text{hash}}, \varepsilon)\)-secure (i.e., \(\text{Adv}\) running in time \(t\), making \(\leq g_{\text{sig}}\) signature calls \& \(\leq g_{\text{hash}}\) hash calls, can forge a new signature w.p. \(\leq \varepsilon\))

where:
\[
\begin{align*}
    t &= t' - \text{poly}(g_{\text{sig}}, g_{\text{hash}}, \varepsilon) \\
    \varepsilon &= (g_{\text{sig}} + g_{\text{hash}}) \cdot \varepsilon'
\end{align*}
\]

**Probabilistic Signature Scheme (PSS)** (a.k.a. RSA-PSS)

[Bellare-Rogaway 96]

RSA-based signature scheme secure in the ROM with tighter security proof.

\[
\begin{align*}
    m, r &\xrightarrow{f} y \xrightarrow{} y^d \mod n \\
    \text{encoding using ROM}
\end{align*}
\]
**El-Gamal Signatures** [1984]

**Note:** The paradigm Enc(Dec(m)) doesn't work for El-Gamal, since El-Gamal is not a trapdoor permutation (it is randomized).

**Scheme:**
- **pp:** prime $p$
  - $g \in \mathbb{Z}_p^*$
  - generator of prime order subgroup $g$ (order $g^{p-1}$).

**KeyGen:**
- $x \in \mathbb{Z}_p^*$
  - $sk = x$
- $y = g^x \mod p$
  - $pk = y$

**Sign** $(pp, sk, m)$:
- Choose $\kappa \leftarrow \mathbb{Z}_p^*$
- Output $(r, s) = \left( g^\kappa \mod p, \frac{h(m) + rx}{\kappa} \mod q \right)$

**Verify** $(pp, pk, m, (r, s))$:
- Check that $0 < r < p$
- Check that $y^{rs} \cdot g^{h(m)s} = r$
Correctness:
\[ y^{r/s} g^{h(m)/s} = g^{xr + h(m)} s = g^k = r \mod p \]

Idea: Generating a signature requires knowledge of \( k \) and a signer that knows \( k \) must know \( sk = x \).

Security:
- Insecure with \( h = \text{identity} \) (exercise).
- Not known to be secure in ROM.
- Secure in ROM if \( h(m) \) is replaced with \( h(m \| r) \).

[Pointcheval–Stern 96]:

Thm: Modified ElGamal is existentially unforgeable against adaptive chosen msg attacks, in ROM, assuming DLP is hard (on avg).

* Rarely used in practice. The following variant is used instead.

Digital Signature Standard
(DSS–NIST91) (a.k.a. Digital Signature Alg)
(DSA)

Public Parameters:
- \( p \) prime, \( q \mid p-1 \)
- \( |p| = 1024 \text{ bits}, \ |q| = 160 \text{ bits} \)
- \( g \) generator of subgroup of \( \mathbb{Z}_p^* \) of order \( q \).
KeyGen: \[ x \leftarrow \mathbb{Z}_p \quad 5x = x \quad 1x = 100 \text{ bits} \]
\[ y = g^x \quad 6y = y \quad 1y = 1024 \text{ bits} \]

Sign_{sk}(m): \[ x \leftarrow \mathbb{Z}_p \]
\[ r = (g^x \mod p) \mod g \quad 1r = 160 \text{ bits} \]
\[ s = \frac{h(m) + rx}{y} \mod g \quad 1s = 160 \text{ bits} \]

Redo if \( r = 0 \) or \( s = 0 \)
Output \( (r, s) \).

Verify_{pk}(m, (r, s)):

- Check \[ 0 < r, s < q \]
- Check \[ y^{r/s} g^{h(m)/s} \pmod{p} \pmod{g} = r \]

Correctness: \[ y^{r/s} g^{h(m)/s} = g^{xr + h(m)} = g^r = r \pmod{p} \pmod{g}. \]

Security: As before, provably secure if \( h(m) \) is replaced with \( h(m \| r) \).