Schnorr signature

$$sP \overset{?}{=} R + hash(R, m)\, P_A$$

Attacker should not be able to create a signature $(R', s') \neq (R, s)$ on any m'.

ECDSA takes only $x(R)$ & doesn't hash in R. So we can flip signs

$$(-s)\,P = -R + h(m)\,P_A$$

$$x\big((-s)\,P\big) = x\big(-R + h(m)\,P_A\big)$$

$R' = R$; $s' = -s$ ⟶ new signature on the same message.

We don't call this an attack but say ECDSA is malleable.

Evil signer can craft a (secret key) so they can swap 2 messages for same (R, s) in ECDSA; revealing m_1 & m_2 will leak a. (see homework)

These signature schemes are fragile for randomness reuse; same R, different m gives a

See Sony PlayStation disaster where a fixed random R was used...

Protection: pick r pseudorandomly (PRF) as $r = hash(k, m)$, where k is a random secret string, which is part of the secret key.

This means we need good randomness at key generation, but not afterwards.

Hash functions are necessary for signature schemes to map the message to a fixed length string in $\{0,1\}^n$.

Symmetric key cryptography

A & B share same key k; want to send m which can be long.

A block cipher encrypts fixed length blocks of data

$$Enc : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$$

$$Enc(k, m) = c$$

(ℓ bits, n bits)

$$Dec : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n \text{ with } Dec(k, Enc(k, m)) = m$$

Enc & Dec are permutations of $\{0,1\}^n$, the choice of permutation is controlled by k.

We want: pseudo-random permutation (PRP)

Currently used block cipher: AES

$n = 128$ (block length), $\ell \in \{128, 192, 256\}$ (key length)

round-based design; number of rounds increases with ℓ

watch Tomer's videos!

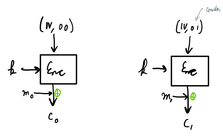
modes of operation deal with chaining blocks of m.

$m = (m_0, m_1, m_2, \ldots)$ each m_i has n bits. Do **not** encrypt block by block as $(c_0, c_1, \ldots) := (Enc(k, m_0), Enc(k, m_1), \ldots)$

This is called Electronic Codebook mode (ECB) and leaks frequency information, that is we have

$$m_i = m_j \Rightarrow c_i = c_j$$

counter mode is simple and secure; see Tomer's videos for alternatives



IV: initialization vector, random string chosen to encrypt m. Receiver sees $(IV, (c_0, c_1, c_2, \ldots)) = (IV, c)$

$$c_i = m_i \oplus Enc(k, (IV, i))$$

(i needs to be binary)

$$m_i = c_i \oplus Enc(k, (IV, i))$$

need to know IV & i

IV & counter need to fit into n bits. Split so that IV is unlikely to ever repeat, while the counter has enough space for the longest message.

E.g. $n = 128$, pick 96 bits for IV, 32 bits for counter.

after 2^{32} blocks of 128-bit m_i, need to pick another IV.

The birthday paradox controls when to change k, because IV will repeat.

Here, after $\approx 2^{48}$ messages of up to 2^{32} blocks.

We DH to re-key (or $k' = hash(k)$)