Attack approaches

modular exponentiation

Cl $\xrightarrow{t_1}$ Server $t_1 + t_2$, $\xleftarrow{t_2}$

time variance: sound, power usage, RF radiation, teletype

0, 1 info leakage,

$n = p \cdot q$

- **Mathematical attacks**: several approaches, all equivalent in effort to factoring the product of two primes. The defense against mathematical attacks is to use a large key size.

side channel attack, gain ~~from~~ info from implementation

- **Timing attacks**: These depend on the running time of the decryption algorithm

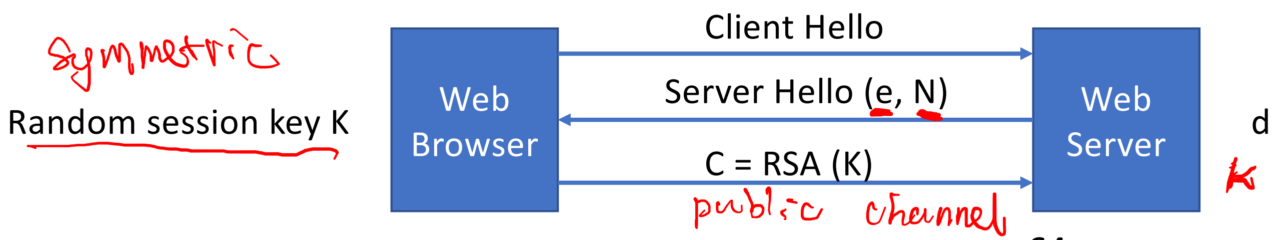
RSA Blinding random $r^e \cdot M^e$ Meta Data Signature

- **Chosen ciphertext attacks**: this type of attacks exploits properties of the RSA algorithm by selecting blocks of data. These attacks can be thwarted by suitable padding of the plaintext, such as PKCS1 V1.5 in SSL

| padding | M | padding |

A simple attack on textbook RSA

SSH or TLS



if (isKeyExist[HashMap, key] == 1)

K_1 = HashMap [K_2]

- Session-key K is 64 bits. View $K \in \{0,\ldots,2^{64}\}$
  - Eavesdropper sees: $C = K^e \pmod N$.

$K = K_1 \cdot K_2$

$K_1 = 1 \cdots C/e$

$C = (K_1 \cdot K_2)^e \bmod N$

- Suppose $\mathbf{K = K_1 \cdot K_2}$ where $K_1, K_2 < 2^{34}$.
  - Then: $\mathbf{C/K_1^{\,e} = K_2^{\,e}} \pmod N$

$K_2^e = C / K_1^e$

$K_2 = 1$

Key index bits HashMap

1. Time complexity $O(1) \cdot 34$

2. # of K_2, 2^{34}

$O(2^{34})$

$K_1 = 1$ $K_1 = 2$

- Build table: $C/1^e, C/2^e, C/3^e, \ldots, C/2^{34e}$. time: 2^{34}

Key		index
K_2	C/K_1^e	$= K_1$
	$C/1^e$	$= 1$
	$C/2^e$	$= 2$
2^{34}		$= 2^{34}$

For $K_2 = 0,\ldots, 2^{34}$ test if $K_2^{\,e}$ is in table. time: $2^{34} \cdot 34$

Brute force

- Attack time: $\approx 2^{40} << 2^{64}$

$O(2^{34}) + O(2^{34} \cdot 34) < 2^{6} \approx 2^{40}$

Take-home exercise – no need to submit

- SW textbook (6th edition) problems: 3.14 & 3.15

n

Homomorphic encryption

- Encryption scheme that allows computation on ciphertexts
  - an extension of public-key encryption scheme that allows anyone in possession of the public key to perform operations on encrypted data without access to the decryption key
- Partially Homomorphic Encryption: Initial public-key systems that allow this for either addition or multiplication, but not both.
  - i.e. RSA
- Fully homomorphic encryption (FHE)

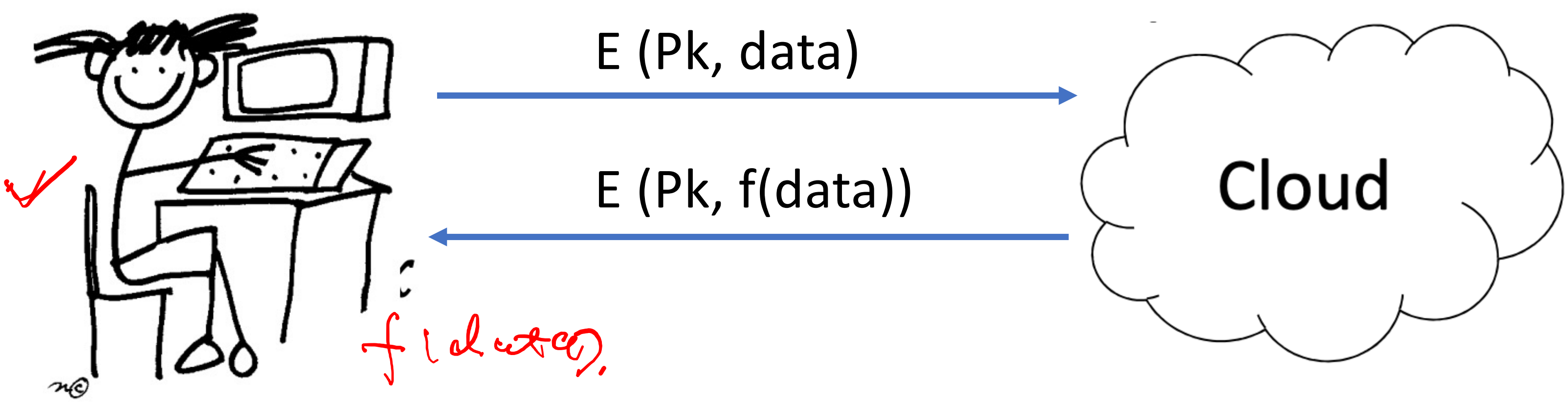
$$
\begin{aligned}
E(m_1)\cdot E(m_2) &= m_1^e \cdot m_2^e \bmod n \\
&= (m_1 \cdot m_2)^e \bmod n \\
&= E(m_1 \cdot m_2)
\end{aligned}
$$

$$
\begin{tikzcd}
M \arrow[r] \arrow[d] & f(M) \\
C \arrow[r] & C' \arrow[u]
\end{tikzcd}
$$

multiplicative.

Application of homomorphic encryption

- One Use case: cloud computing LLM.
  - A weak computational device Alice (e.g., a mobile phone or a laptop) wishes to perform a computationally heavy task, beyond her computational means. She can delegate it to a much stronger (but still feasible) machine Bob (the cloud, or a supercomputer) who offers the service of doing so. The problem is that Alice does not trust Bob, who may give the wrong answer due to laziness, fault, or malice.



RSA reading materials

- A Method for Obtaining Digital Signatures and Public-Key Cryptosystems