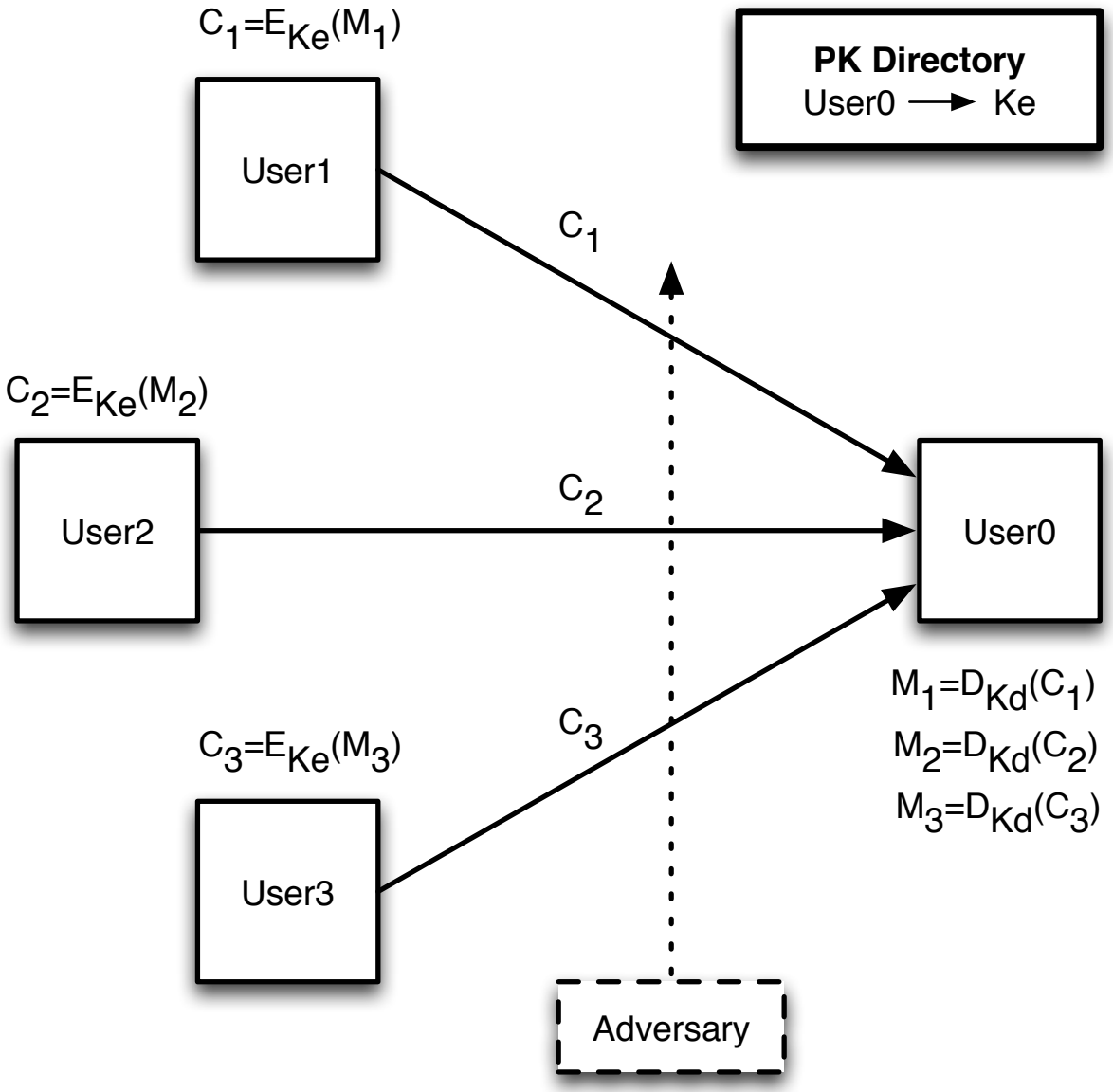
Digital Signatures

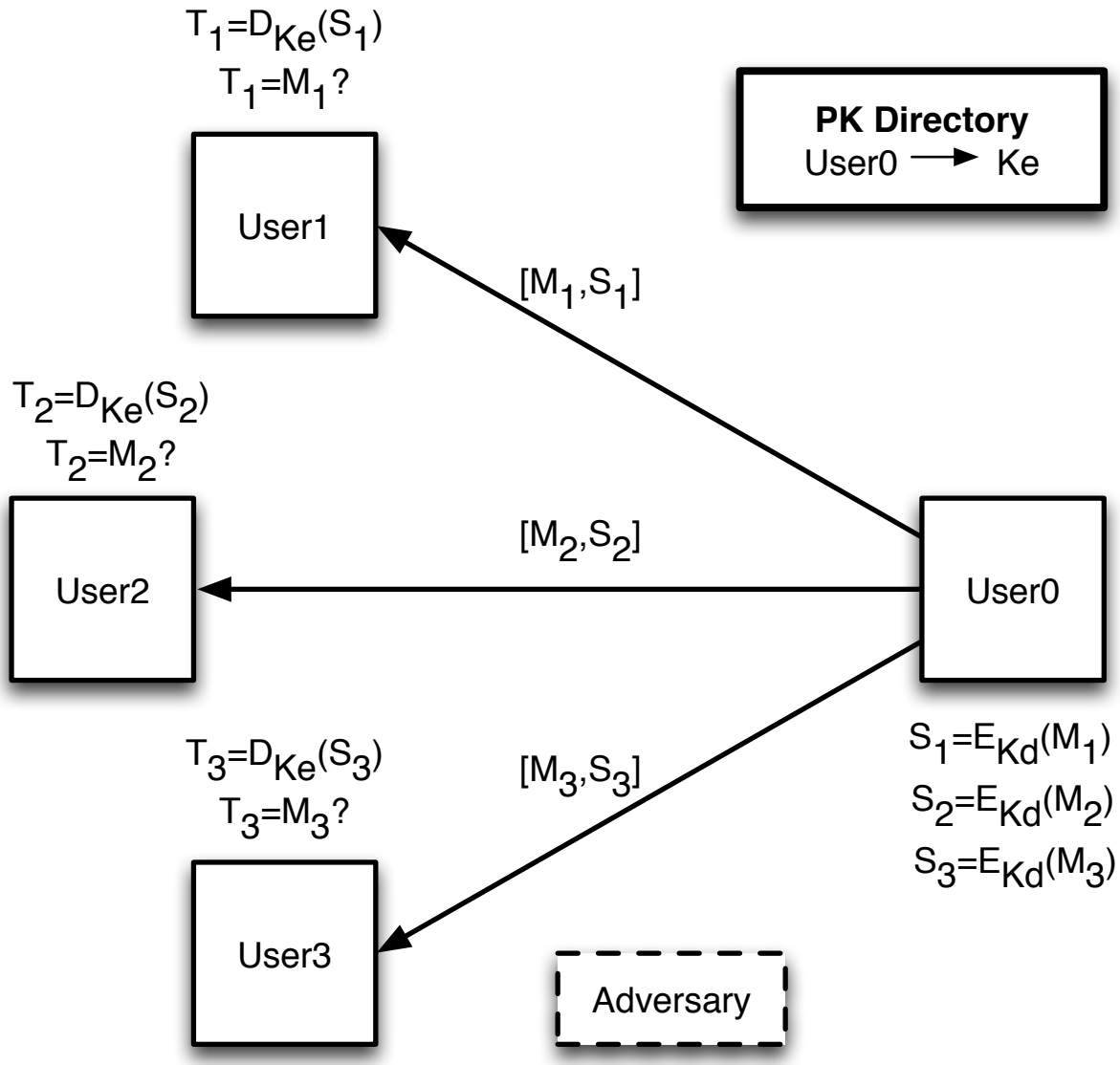
Digital Signatures

- A digital signature or digital signature algorithm is a mathematical method for demonstrating the authenticity of a digital message or document
- A valid digital signature gives a recipient reason to believe that the message was created by a known sender (authentication) such that he/she cannot deny sending it (non-repudiation) and that the message was not altered in transit (integrity)
- Digital signatures are commonly used for software distribution, financial transactions, and in other cases where it is important to detect forgery or tampering
- A public-key encryption algorithm is also a digital signature algorithm, the most notable example being the RSA algorithm

Public-Key Encryption



Digital Signatures



Digital Signatures

- In 1976, Diffie and Hellman first described the concept of a digital signature scheme, and they conjectured that such methods exist
- The RSA algorithm can be used as a public-key encryption method and as a digital signature algorithm
- However, the plain RSA signatures have certain security problems
- Other digital signature algorithms have been developed after the RSA: Lamport signatures, Merkle signatures, and Rabin signatures
- Several more digital signature algorithms followed up, and are in use today: ElGamal, the Digital Signature Algorithm (DSA), the elliptic curve DSA (ECDSA)

Plain RSA Signatures

- User A has an RSA public key (n, e) and private key (n, d)
- User A creates a message $M < n$, and **encrypts the message using the private key** to obtain the signature S as

$$S = M^d \pmod{n}$$

and sends the message (plaintext) and the signature $[M, S]$ to User B

- User B receives $[M, S]$, obtains User A's public key from the directory, and **decrypts the signature using the public key**:

$$T = S^e \pmod{n}$$

If $T = M$, then the User B decides that the signature S on the message M was created by User A

Plain RSA Signatures

- Plain RSA signatures have several problems to be used directly as a signature scheme in practice
- The message length is limited to the modulus length
- Longer messages cannot be directly signed
- Legitimate signatures can be used to create **forged** signatures

I forged your RSA signature

Forged RSA Signatures

- Consider that $[M, S]$ is a legitimate pair of message and signature, created by the owner of the public and private key pair such that $S = M^d \pmod{n}$ and $M = S^e \pmod{n}$
- The pair $[M^2 \bmod n,\ S^2 \bmod n]$ also verifies:

$$(S^2)^e = (S^e)^2 = M^2 \pmod{n}$$

- It appears that $[M^2 \bmod n,\ S^2 \bmod n]$ is a legitimate signature
- Furthermore, we can take two message and signature pairs: $[M_1, S_1]$ and $[M_2, S_2]$, and forge a new signature as

$$[M', S'] = [M_1 \cdot M_2 \bmod n,\ S_1 \cdot S_2 \bmod n]$$

Forged RSA Signatures

- For any two integers i and j, we can create as many forged signatures as we need from two given legitimate signatures $[M_1, S_1]$ and $[M_2, S_2]$

$$[M',\ S'] = [M_1^i \cdot M_2^j \bmod n,\ S_1^i \cdot S_2^j \bmod n]$$

- Our only hope is that the new message obtained from multiplying existing messages is no longer a "real" message

$$M' = M_1^i \cdot M_2^j \pmod n$$

since messages generally have some structure and redundancy in them, for example, ASCII encoding, sound or picture formats, etc

Signing Structured Messages

- Since digital data are in general highly structured and have certain formats, modular arithmetic operations on them will likely produce numbers that will fail the structure
- Take a minute of a Mozart violin concerto, and represent it as a large integer m (analog to digital conversion followed up by some quantization technique)
- What are the chances that $m^2 \pmod{n}$ for any n will produce anything that resembles like a music: Nil
- Or, take some sort of digital contract, for example, a rent contract, and represent it as a large integer
- What are the chances that $m^2 \pmod{n}$ for any n will produce anything that resembles like a digital document: Nil
- Therefore, we are pretty safe that forged plain signatures will not produce believable data

Signing Random Messages

- On the other hand, signatures on random data are easily forgeable
- If M is random and $[M, S]$ is a valid message and signature pair, then $[M^2, S^2]$ will appear to be a valid message and signature pair, indistinguishable from another $[M, S]$ since both M and M^2 are random
- Therefore, plain RSA signatures on random messages should be used with extreme caution
- Some special cryptographic protocols may require signing random numbers
- In such protocols, there are assumptions that parties executing the protocol are "honest", i.e., they do not attempt to forge signatures from actual, valid ones

Digital Signatures for Long Digital Data

- There is another significant practical issue with plain RSA signatures
- The fact the the RSA signature method limits the size of the message to be signed to the size of the modulus is too restrictive for almost all business applications
- Timing and hardware resource constraints and security arguments keep the modulus size around 1024 bits for now, or perhaps 2048 bits
- This is only up to 256 bytes of message, or 256 characters of ASCII, i.e., about 4 lines of text
- However, most documents are several pages or tens of pages long
- Therefore, it is imperative to have a digital signature practice for very long (unlimited length) messages

Digital Signatures and Hashing

- Instead of encrypting M with the private key, we encrypt $H(M)$: the hash of M

$$h = H(M) \quad \rightarrow \quad S = h^d \pmod{n} \quad \rightarrow \quad [M, S]$$

- The receiving party verifies the message and signature pair $[M, S]$ using

$$h = H(M) \quad \rightarrow \quad T = S^e \pmod{n} \quad \rightarrow \quad T \stackrel{?}{=} h$$

- The cryptographic hash function $H(\cdot)$ is a publicly available function, and does not involve a secret key