CSE 509 Lecture Notes for April 10, 2007
By: Gabriel Sanchez

The following is some background information needed for the Terra paper:

Public Key Cryptography:

Each person has a public key, which can be given to anyone, and a private key, which is a secret.

Alice has the following:

\( P_A \) – Public Key  \( S_A \) – Private (Secret) Key

In order to encrypt a message to Alice,

\[ C = E \left( P_A, M \right) \]

In which \( C \) is the ciphertext and \( M \) is the message. To read a message (which only Alice can do with her secret key):

\[ M = D \left( S_A, C \right) \]

RSA:

To generate a public/private key pair, Alice does:

1. Pick prime numbers \( p \) and \( q \) (usually large numbers)
2. \( N = p \times q \), and \( \phi(N) = (p-1)(q-1) \)
3. Find \( e \) and \( d \) such that \( e \times d = 1 \mod \phi(N) \)
4. \( P_A = (e, N) \) and \( S_A = (d, N) \)
Then, using Fermat’s Little Theorem,

\[ E ( ( e , N ) , M ) = M^e \mod N = C \]
\[ D ( ( d , N ) , C ) = C^d \mod N = M^{ed} \mod N = M \]

**Example of RSA:**

\( p = 7 \) and \( q = 5 \) [Step 1]

So, \( N = 7 \times 5 = 35 \), \( \phi ( N ) = ( 7 - 1 ) \times ( 5 - 1 ) = 6 \times 4 = 24 \) [Step 2]

If \( e = 7 \) and \( d = 7 \) (note: in real systems, \( e \) and \( d \) are **not** the same) [Step 3]

- verified by calculating \( e \times d = 1 \mod \phi ( N ) \):
  
  \[ 7 \times 7 = 49 \0 49 \mod 24 = 1 \]

Then, \( P_A = ( 7 , 35 ) \) and \( S_A = ( 7 , 35 ) \) [Step 4]

Now, if the original message is say \( M = 2 \), the following illustrates how to perform the encryption/decryption process:

\[ E = ( P_A , 2 ) = 2^7 \mod 35 = 23 \]
\[ D = ( S_A , 23 ) = 23^7 \mod 35 = 2. \text{ Since } D = M = 2, \text{ success.} \]

**Public Key Signatures:**

Only one person can generate a signature, but anyone can verify it.

Idea: Each user has a private signing key and a public verification key, \( S_A \) and \( P_A \) respectively.

\[ S = \text{Sign} ( S_A , M ) \text{ in which } M \text{ can be considered an unforgeable contract} \]

\[ \text{Verify} = ( P_A , M , S ) = \text{Valid or Invalid} \text{ if } S \text{ is the same message signed.} \]

Similar to RSA encryption/decryption process, except that \( e \) and \( d \) exponents are switched:

\[ \text{Sign} ( ( d , N ) , M ) = M^d \mod N = S \]
\[ \text{Verify} ( ( e , N ) , M , S ) = \text{Valid if } S^e = M \mod N \]
Signing Long Messages

Use a hash function

\[ S = \operatorname{Sign}(S_A, \text{hash}(M)) \quad h: \{0, 1\}^* \rightarrow \{0, 1\}^n \]

Strong collision resistance: hash has a strong collision resistance if it is hard to find \( x \) and \( y \) such that \( \text{hash}(x) = \text{hash}(y) \) and \( x \) is not equal to \( y \).

Certificates:

To verify Alice’s signature, Bob needs to know \( P_A \). Suppose Alice and Bob trust Charlie.

Alice: somehow \quad Charlie: \quad Bob:
\[ P_A, S_A, \text{and } P_C (\text{somehow}) \quad \overset{\circ}{\longrightarrow} \quad S_C \quad \overset{\circ}{\longrightarrow} \quad P_C \]

\[ C_A^C = \operatorname{Sign}(S_C, \text{“Alice’s public key is } P_A \text{”}) \]

If Frank sends a message to Bob, Bob has to trust all parties leading to Frank.
Terra / TVMM (Trusted Virtual Machine Monitor)

A system for running programs on a program with the following assumptions:

- The owner of the computer is malicious
- “root_secure” system needed
- Applications (to be protected)
  - Network Games – to prevent players from cheating using various techniques such as making walls semi-transparent
  - Mouse/Music Players – using Digital Rights Management (DRM) to prevent copying and redistributing of music files
  - Reverse Engineering (protecting certain information) by “security through obscurity” – hide the location of a key used for DRM from user.

Trusted Virtual Machine Monitor

Each guest “thinks” it is running on its own computer

- VMM exposes Hardware interface to guest VMs
- Strong isolation between VMs (different memory, CPU, Disk, Network, etc.)
Network games bring up the idea of:

**Remote Attestation:**

Goal: To prove to a remote party that I am running a certain set of software.

Example: Apps running, OS, VMM, Bootloader, BIOS, etc.

(more on Terra in the next lecture)