6.857 Recitation 7: MITM Attacks, Digital Signatures Review

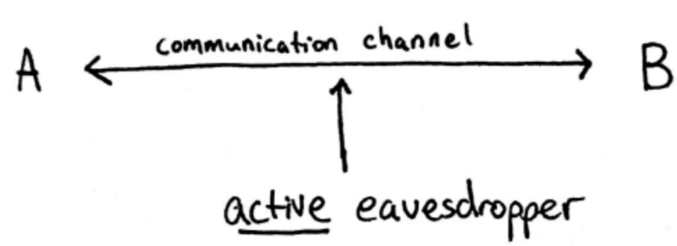
TA: Sean Fraser

Friday March 22nd, 2019

Agenda

- Reminder: Project proposals due tonight!
- Man-In-The-Middle Attacks (MITM)
  - Diffie-Hellman (DH) Key Exchange Example
- Digital Signatures Review
  - Definition
  - Hash & Sign
  - El-Gamal Signature Scheme
  - Digital Signature Standard (DSS) / DSA
- Questions

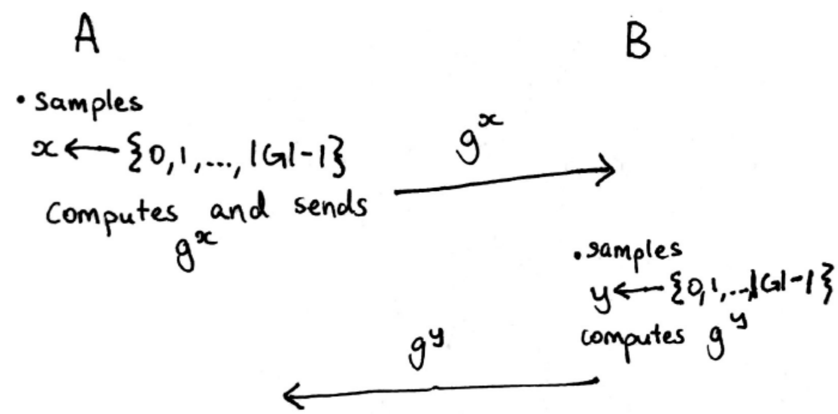
1 Man-In-The-Middle Attacks



We will illustrate an example of a Man-In-The-Middle attack using the textbook Diffie-Hellman (DH) Key Exchange. Suppose we have a communication channel between Alice (A) and Bob (B) with an active eavesdropper (Eve, or E) as shown. In class we showed this setup with a passive eavesdropper, and we will show why an active eavesdropper is problematic.

Recall: DH Key Exchange

- G is a finite cyclic group, with generator g.
  - $G = \{g^0, g^1, ..., g^{|G|-1}\}$
  - G and g are fixed and public



- A and B compute $K = g^{xy} = (g^x)^y = (g^y)^x$
- Relies on DDH - Decisional Diffie Hellman Assumption: $(g^x, g^y, g^{xy}) \approx_c (g^x, g^y, g^z)$

  Given g^x and g^y, cannot distinguish between g^{xy} and g^z with probability $> \frac{1}{2} + \lambda$, where $u \leftarrow \{0, 1, ..., |G| - 1\}$ (randomly drawn).

  Note: confer with CDH, Computational Diffie Hellman assumption, in Lecture 9, which is less strong.

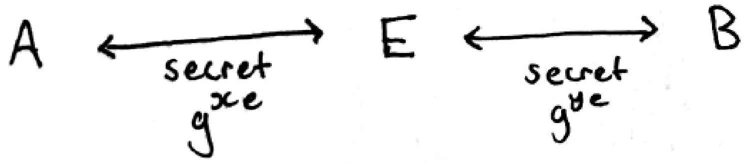
  Assuming DDH, Diffie Hellman is secure under a *passive* adversary.

  Problem: Totally insecure to an *active* eavesdropper.

  Man-in-the-Middle Attack (MITM): active eavesdropper can intercept and relay messages in between Alice and Bob. In the DH key exchange for example, this means the adversary can establish a different key with each of A and B separately, using the DH key exchange, tricking Alice and Bob that Eve is the other person respectively when she is really not. This might work as shown below, with Eve intercepting each of g^x from Alice and g^y from Bob and sending g^e to both. This gives Eve full power to encrypt and decrypt messages between Alice and Bob, and change them how she likes.

  Problem: Authenticity. A and B have no way of verifying the "identity" of the other.

  Potential solution: Digital Signatures.



2 Digital Signatures

- **Idea:** each user has a pair of keys (PK, SK). PK is the public key, SK is the secret key.
- **Want:** one person to be able to sign, and everyone to able to verify the signature (that it came from the source it says).

  $\implies$ SK to sign, PK to verify.
- **Recall Definition:** Digital Signature Schemes
  - $\underline{Keygen(1^\lambda)} \rightarrow (PK, SK)$
  - $\underline{Sign(SK, m)} \rightarrow \sigma_{SK}(m)$ (may be randomized)
  - $\underline{Verify(PK, m, \sigma)} \rightarrow True/False$

Intuitively, a signature scheme is correct (different to secure) if for all m, we have

$$Verify(PK, m, Sign(SK, m)) = True$$

Security: against adaptive chosen message attacks (game-based definition). This is also called existential unforgeability.

1. Challenger generates $(PK, SK) \leftarrow Keygen(1^\lambda)$
2. Adversary gets oracle access to sign $(SK, \bullet)$ i.e., adversary get signatures to sequence of messages of his choice: $m_1, ..., m_q$ such that $q = poly(\lambda)$. Note that m_i can depend on the previous signatures given ("adaptive"). For notation, let $\sigma_i = sign(SK, m_i)$.
3. Adversary outputs a pair (m, σ^*)

The adversary wins if:

1. $Verify(PK, m, \sigma^*) = 1$
2. $m \notin \{m_1, ..., m_q\}$

The signature scheme is <u>secure</u> if $Pr[\text{Adv Wins}] \leq negl(\lambda)$ i.e. a negligible function of λ (For an exact definition, if you are interested, see the Katz and Lindell textbook or the lecture notes, but this should be enough. Also note that there are notions of *strong* and *weak* security against adaptive chosen message attacks, where technically the definition above is *weak* security and *strong* security against adaptive chosen message attacks or *strong* existential unforgeability is where the adversary is allowed to output a signature for a message he has already seen, but the new signature has to be different. The above definition is all that is needed for the class though, and we will not be distinguishing between the two).

- First idea: we want to use a deterministic public key encryption scheme as a signature scheme.

  $Sign(SK, m) = Dec(SK, m)$

  $Verify(PK, m, \sigma) = 1 \Leftrightarrow Enc(PK, \sigma) = m$

  Note, this is kind of opposite to how we do encryption in PK cryptography, but we need the signing function to use the secret key.

- Problem: As shown in class e.g. with RSA (a trapdoor function - easy to compute one way, but hard to invert)

  $Sign(SK, m) = m^d \mod n$

  But can easily sign $m^2 \mod n = (m^d)^2 \mod n \rightarrow$ <u>insecure</u>.

  If this is confusing, refer to the lecture notes for RSA for how we set up the RSA parameters and RSA signatures.

- <u>Hash & Sign Paradigm</u>

  e.g. for RSA, with a hash function h:

  $Sign((SK, h), m) = (h(m))^d \mod n$

  $Verify((PK, h), m, \sigma) = 1$ IFF $\sigma^e = h(m) \mod n$

  Security depends on h, need collision resistance at least. (Note identity function is collision resistant, but not secure).

  Secure if h modeled as Random Oracle (ROM) (not secure if $h(m)^d \rightarrow_{\text{easy}} h(m^2)^d$).

  Advantages of hash & sign: enhances security, more efficient to work with smaller fixed-length output of hash function, flexibility.

2.1 El Gamal Signatures (Review)

- Can't use same method to convert encryption scheme to signature scheme like RSA, since El Gamal is randomized.

- Public Parameters (PP): prime p, generator $g \in \mathbb{Z}_p^*$ of prime order subgroup q, such that $q | p - 1$. For example, g could be a $QR \neq 1, QR \in \mathbb{Q}_p^*$, with $p = 2q + 1$.

- KeyGen: Sample randomly $x \leftarrow \mathbb{Z}_q$

  $y = g^x \mod p$

  $SK = x$, $PK = g^x = y$

  Security of secret key x relies on the Discrete Log Assumption (i.e. given $y = g^x \mod p$ it is computationally infeasible to find x given y and g (and p)).

- $\underline{Sign(PP, SK, m)}$: Sample randomly $k \leftarrow \mathbb{Z}_q^*$

  Let $r = g^k \mod p$

  Output $(r, s) = (g^k \mod p, \frac{h(m)+r\cdot x}{k} \mod q)$

- $\underline{Verify(PP, PK, m, (r, s))}$:

  Check that $0 < r < p$

  Check that $y^{\frac{r}{s}} \cdot g^{\frac{h(m)}{s}} = r$

- Correct since $y^{\frac{r}{s}} \cdot g^{\frac{h(m)}{s}} = g^{\frac{x\cdot r+h(m)}{s}} = g^k = r \mod p$

- Pointcheval-Stern (1996)

  replace $h(m)$ with $h(m||r)$

  $\implies$ now secure against adaptive chosen message attacks, assuming ROM (previously unsure)

2.2 DSS and DSA (Review)

- Digital Signature Standard (DSS) - DSA (Digital Signature Algorithm) meets this standard set and developed by NIST.

- variant of El Gamal Signatures

- much faster / efficient due to a few key differences (works in subgroup of order q, as opposed to $\mod p$ - order p)

  about 6 times smaller signatures (6 times faster)

- Key differences: (for full specification, see Lecture Notes on Digital Signatures)

  $|p| = 1024$ bits, $|q| = 160$ bits

  $r = (g^k \mod p) \mod q$, $|r| = 160$ bits, $|s| = 160$ bits.

  All operations in 160-bit subgroup (as opposed to full $\mathbb{Z}_p^*$, which would have been 1024 bits)

  Same provable level of security if $h(m||r)$ is used.

- key note:

- importance of randomly selected k, where $k \leftarrow \mathbb{Z}_q^*$
- if k reused for different messages m, one could solve for x (the secret key) (to be shown on problem set problem).
- if k different for same m, it should be random and unknown. Any relation between the two k's allows to solve for x.
- Therefore, it is critical that k is properly random and unique. Also q has to be large enough to prevent brute-force attacks.
- e.g. Sony 2010: someone cracked x in the DSA algorithm that Sony was using, since they failed to generate random $k's$ for each signature.