15: Network Security Basics

Last Modified: 4/21/2003 8:30:27 PM

Importance of Network Security
- Society is becoming increasingly reliant on the correct and secure functioning of computer systems
  - Medical records, financial transactions, etc.
- It is our jobs as professional computer scientists:
  - To evaluate the systems we use to understand their weaknesses
  - To educate ourselves and others to be wise network consumers
  - To design networked systems that are secure

Types of attacks
- What are we worried about?
  - Passive:
    - Interception: attacks confidentiality, a.k.a., eavesdropping, "man-in-the-middle" attacks.
    - Traffic Analysis: attacks confidentiality, or anonymity. Can include tracevork on a network, CRT radiation.
  - Active:
    - Interruption: attacks availability
    - Modification: attacks integrity
    - Fabrication: attacks authenticity

Fundamentals of Defense
- What can we do about it?
  - Restricted Access
    - Restrict physical access, close network ports, isolate from the Internet, firewalls, NAT gateways, switched networks
  - Monitoring
    - Know what normal is and watch for deviations
  - Heterogeneity/Randomness
    - Variety of Implementations, Random sequence numbers, Random port numbers
  - Cryptography

Cryptography
- The most widely used tool for securing information and services is cryptography.
- Cryptography relies on ciphers: mathematical functions used for encryption and decryption of a message.
  - Encryption: the process of disguising a message in such a way as to hide its substance.
  - Ciphertext: an encrypted message
  - Decryption: the process of returning an encrypted message back into plaintext.

<table>
<thead>
<tr>
<th>Plaintext</th>
<th>Ciphertext</th>
<th>Original</th>
</tr>
</thead>
</table>
| Encryption | Decryption | }
What makes a good cipher?

substitution cipher: substituting one thing for another
monoalphabetic cipher: substitute one letter for another
plaintext: abedefghijklmnopqrstuvwxyz
ciphertext: mnbvcsadfhjklpoiuytrewq

E.g.: Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc

Q: How hard to break this simple cipher?:
• brute force (how hard?)
• other?

Ciphers

The security of a cipher (like a substitution cipher) may rest in the secrecy of its restricted algorithm.
• Whenever a user leaves a group, the algorithm must change.
• Can't be scrutinized by people smarter than you.
• But, secrecy is a popular approach :(

Modern cryptography relies on secret keys, a selected value from a large set (a key space), e.g., a 1024-bit number. $2^{1024}$ values!
• Security is based on secrecy of the key, not the details of the algorithm.
• Change of authorized participants requires only a change in key.

Keys: Symmetric vs Asymmetric

The most common cryptographic tools are

Symmetric key ciphers
• Use same key to encrypt and decrypt
• One key shared and kept secret
• DES, 3DES, AES, Blowfish, Twofish, IDEA
• Fast and simple (based on addition, masks, and shifts)
• Typical key lengths are 40, 128, 256, 512

Asymmetric key ciphers
• Pair of keys: one encrypts and another decrypts
• One key (the private key) must be kept secret; the other key (the public key) can be freely disclosed
• RSA, El Gammal
• Slow, but versatile (usually requires exponentiation)
• Typical key lengths are 512, 1024, 2048

Session Keys

Symmetric key algorithms are faster than asymmetric key algorithms
• Often asymmetric key cryptography used to exchange a shared secret key
• This key called a symmetric session key is then used to encrypt this conversation with symmetric key cryptography
• Each new conversation would use a different session key
• Other benefits (In addition to efficiency)
  • session keys also reduce the key exposure or amount of encrypted text that could be collected to aid in analysis
  • If session key compromised only get info in the last session

Symmetric key crypto: DES

DES: Data Encryption Standard
• US encryption standard [NIST 1993]
• 56-bit symmetric key, 64 bit plaintext input
  • initial permutation
  • 16 identical "rounds" of function application, each using different 48 bits of key
  • final permutation
• How secure is DES?
  • DES Challenge: 56-bit-key-encrypted phrase decrypted (brute force) in a little over 22 hours (1999 DES Challenge III)
  • no known "backdoor" decryption approach
• making DES more secure
  • use three keys sequentially (3-DES) on each datum
  • use cipher-block chaining

Public key encryption algorithms

Two inter-related requirements:
• need a decryption function $d_B(\cdot)$ and an encryption function $e_B(\cdot)$ such that $d_B(e_B(m)) = m$ and $e_B(d_B(m)) = m$
• need public and private keys for $d_B(\cdot)$ and $e_B(\cdot)$
RSA

- Ronald L. Rivest, Adi Shamir and Leonard M. Adleman
- Won 2002 Turing award for this work!
- Want a function $e$ that is easy to do, but hard to undo without a special decryption key
- Based on the difficulty of factoring large numbers (especially ones that have only large prime factors)

RSA in a nutshell

1. Choose two large prime numbers $p$, $q$.
   (e.g., 1024 bits each)
2. Compute $n = pq$, $z = (p-1)(q-1)$
3. Choose $e$ (with $e \cdot n$) that has no common factors with $z$. (e, z are "relatively prime").
4. Choose $d$ such that $ed - 1$ is exactly divisible by $z$
   (in other words: $ed \mod z = 1$).
5. Public key is $(n, e)$. Private key is $(n, d)$.

Why? (Will hint at)

How? (Won’t discuss)

RSA: Encryption, decryption

0. Given $(n, e)$ and $(n, d)$ as computed above
1. To encrypt bit pattern (message), $m$, compute
   $c = m^e \mod n$ (i.e., remainder when $m^e$ is divided by $n$)
2. To decrypt received bit pattern, $c$, compute
   $m = c^d \mod n$ (i.e., remainder when $c^d$ is divided by $n$)

   Magic happens
   $m = (m^e \mod n)^d \mod n$

RSA: small example

$e=5$ (so $e, z$ relatively prime),
$d=29$ (so $ed - 1$ exactly divisible by $z$).

encrypt:

<table>
<thead>
<tr>
<th>letter</th>
<th>$m$</th>
<th>$m^e$</th>
<th>$c = m^e \mod n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>l</td>
<td>12</td>
<td>1524832</td>
<td>17</td>
</tr>
</tbody>
</table>

decrypt:

<table>
<thead>
<tr>
<th>$c$</th>
<th>$c^d$</th>
<th>$m = c^d \mod n$</th>
<th>letter</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>12</td>
<td>481868232339728691816253637029</td>
<td>l</td>
</tr>
</tbody>
</table>

RSA: Why?

$m = (m^e)^d \mod n$

Number theory result: If $p, q$ prime, $n = pq$, then
$x \mod n = x \mod (p-1)(q-1)$

$(m^e)^d \mod n = m^{ed} \mod n$

If it were easy to factor $n$ into $p$ and $q$ then we would be in trouble!

Reversible

- What the private key encrypts the public key decrypts
- What the public key encrypts the private key decrypts
Practical matters

- Big primes like 5 and 7 (∘) already generated big numbers like 481968572106750915091411825223072000
  - What would happen with 1024 bit keys?
  - Costly operations!
- Finding big primes?

Storing your keys

- For both symmetric and asymmetric cryptography how do you store the keys?
  - Typical key lengths are 512, 1024, 2048
- Can’t exactly memorize it
- Ok to store in on your computer? In a shared file system? No!
- Normally stored in a file encrypted with a pass phrase
- Pass phrase != your key

Using Cryptography

- Secrecy/Confidentiality: ensuring information is accessible only by authorized persons
  - Traditionally, the primary objective of cryptography.
  - E.g. encrypting a message
- Authentication: corroboration of the identity of an entity
  - Allows receivers of a message to identify its origin
  - Makes it difficult for third parties to masquerade as someone else
  - E.g., your driver’s license and photo authenticates your image to a name, address, and birth date.

Uses of Cryptography

- Integrity: ensuring information has not been altered by unauthorized or unknown means
  - Integrity makes it difficult for a third party to substitute one message for another.
  - It allows the recipient of a message to verify it has not been modified in transit.
- Nonrepudiation: preventing the denial of previous commitments or actions
  - Makes it difficult for the originator of a message to falsely deny later that they were the party that sent the message.
  - E.g., your signature on a document.

Friends and enemies: Alice, Bob, Trudy

- Well-known in network security world
- Bob, Alice want to communicate “securely”
- Trudy, the “intruder” may intercept, delete, add messages
Digital Signatures

Cryptographic technique analogous to handwritten signatures.
- Sender (Bob) digitally signs document, establishing he is document owner/creator.
- Verifiable, nonforgeable: recipient (Alice) can verify that Bob, and no one else, signed document.

Simple digital signature for message \( m \):
- Bob encrypts \( m \) with his private key \( d_B \), creating signed message, \( d_B(m) \).
- Bob sends \( m \) and \( d_B(m) \) to Alice.

Alice thus verifies that:
- Bob signed \( m \).
- No one else signed \( m \).
- Bob signed \( m \) and not \( m' \).

Non-repudiation:
- Alice can take \( m \), and signature \( d_B(m) \) to court and prove that Bob signed \( m \).

Message Digests

Computationally expensive to public-key-encrypt long messages

Goal: fixed-length, easy to compute digital signature, "fingerprint"
- Apply hash function \( H \) to \( m \), get fixed size message digest, \( H(m) \).

Hash function properties:
- Many-to-1
- Produces fixed-size message digest (fingerprint)
- Given message digest \( x \), computationally infeasible to find \( m \) such that \( x = H(m) \)
- Computationally infeasible to find any two messages \( m \) and \( m' \) such that \( H(m) = H(m') \)

Digital signature = Signed message digest

Bob sends digitally signed message:
- Alice verifies signature and integrity of digitally signed message.

Internet checksum would make a poor message digest:
- Too easy to find two messages with same checksum.

MD5 hash function widely used:
- Computes 128-bit message digest in 4-step process.
- Arbitrary 128-bit string \( x \), appears difficult to construct \( m \) whose MD5 hash is equal to \( x \).
- SHA-1 is also used.
- US standard
- 160-bit message digest
Authentication

**Goal:** Bob wants Alice to "prove" her identity to him

**Protocol ap1.0:** Alice says "I am Alice"

Failure scenario??

Authentication: another try

**Protocol ap3.0:** Alice says "I am Alice" and sends her secret password to "prove" it.

Failure scenario?

Authentication: yet another try

**Protocol ap3.1:** Alice says "I am Alice" and sends her encrypted secret password to "prove" it.

Failure scenario? Trudy can't decrypt password But can still replay it

Authentication: ap5.0

**Problem:** how do Bob, Alice agree on key?

**ap4.0:** use nonce, public key cryptography

What proves \( e_a \) is Alice's public key?

**ap5.0:** use nonce, public key cryptography

**Man (woman) in the middle attack:** Trudy poses as Alice (to Bob) and as Bob (to Alice)

Need "certified" public keys
Trusted Intermediaries

**Problem:** How do two entities establish shared secret key over network?

**Solution:**
- Trusted key distribution center (KDC) acting as intermediary between entities

**Problem:** How do two entities establish shared secret key over network?

**Solution:**
- Trusted key distribution center (KDC) acting as intermediary between entities

Key Distribution Center (KDC)

- Alice, Bob need shared symmetric key.
- KDC: server shares different secret key with each registered user.
- Alice, Bob know own symmetric keys, $K_A$-KDC, $K_B$-KDC, for communicating with KDC.

Certification Authorities

- Certification authority (CA) binds public key to particular entity.
- Entity (person, router, etc.) can register its public key with CA.
- Entity provides "proof of identity" to CA.
- CA creates certificate binding entity to public key.
- Certificate digitally signed by CA.
- Public key of CA can be universally known (on billboard, embedded in software) - unless have to change because private key compromised.

Establishing Trust

- Is the problem of establishing "trust" with a key authority or certification authority the same as establishing "trust" with anyone else?
- Public Key: CA can put their public key on a bulletin board but how do you convince them that your public key really is your public key?
- Problem is the same!!
- Use out of band means
- BUT!!! Once you establish trust with them you can use that to bootstrap trust with others