15-441 Computer Networking

Lecture 27 – Security and Cryptography

Ed Bardsley
(Special Thanks to John Heffner)

Outline

- Security motivations
- Private key cryptosystems
- Public key cryptosystems
- Application level protocols
- IPsec

Security Threats

- Impersonation
  - Pretend to be someone else to gain access to information or services
- Insecrecy
  - Eavesdrop on data over network
- Corruption
  - Modify data over network
- Repudiation
  - Deny sending a message
- Break-ins
  - Take advantage of implementation bugs
- Denial of Service
  - Flood resource to deny use from legitimate users

Three Levels of Defense

- Firewalls
  - Filtering “dangerous” traffic at a middle point in the network
  - Covered next lecture
- Network level security (e.g. IPsec)
  - Host-to-host encryption and authentication
  - Can provide security without application knowledge
  - Cannot always protect applications from each other
- Application level security
  - True end-to-end security
  - Requires extra effort per application
  - Libraries help, like SSL/TLS

Private Key Cryptosystems

- Finite message domain M, key domain K
- Key $k \in K$
  - Known by all parties
  - Must be secret
- Encrypt: $E: M \times K \rightarrow M$
  - Plaintext m_p to ciphertext m_c as $m_c = E(m_p, k)$
- Decrypt: $D: M \times K \rightarrow K$
  - $m_p = D(m_c, k) = D(E(m_p, k), k)$
- Cryptographic security
  - Given m_c, hard to determine m_p or k
  - Given m_c and m_p, hard to determine k

One Time Pad

- Messages
  - n-bit strings $[b_1,\ldots,b_n]$
- Keys
  - Random n-bit strings $[k_1,\ldots,k_n]$
- Encryption/Decryption
  - $c = E(b, k) = b \oplus k = [b_1 \oplus k_1, \ldots, b_n \oplus k_n]$
    - $\oplus$ denotes exclusive or
  - $b = D(b, k) = c \oplus k = b \oplus k \oplus k = b \oplus [1, \ldots, 1] = b$
- Properties
  - Provably unbreakable if used properly
  - Keys must be truly random
  - ***Must not*** be used more than once
  - Key same size as message

Simple Permutation Cipher

- Messages
  - n-bit strings $[b_1,\ldots,b_n]$
- Keys
  - Permutation π of n
  - Let $\sigma = \pi^{-1}$
- Encryption/Decryption
  - $E([b_1,\ldots,b_n], \pi) = [b_{\pi(1)},\ldots,b_{\pi(n)}]$
  - $D([b_1,\ldots,b_n], \pi) = [b_{\sigma(1)},\ldots,b_{\sigma(n)}]$
- Properties
  - Cryptanalysis possible
  - Only small part of plaintext and key used for each part of ciphertext

Data Encryption Standard (DES)

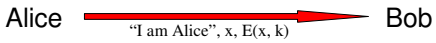
- History
  - Developed by IBM, 1975
  - Modified slightly by NSA
  - U.S. Government (NIST) standard, 1977
- Algorithm
  - Uses 64-bit key, really 56 bits plus 8 parity bits
  - 16 "rounds"
    - 56-bit key used to generate 16 48-bit keys
    - Each round does substitution and permutation using 8 S-boxes
- Strength
  - Difficult to analyze
  - Cryptanalysis believed to be exponentially difficult in number of rounds
  - No currently known attacks easier than brute force
  - But brute force is now (relatively) easy

Other Ciphers

- Triple-DES
  - DES three times
    - $m_c = E(D(E(m_p, k_1), k_2, k_3)$
  - Effectively 112 bits
  - Three times as slow as DES
- Blowfish
  - Developed by Bruce Schneier circa 1993
  - Variable key size from 32 to 448 bits
  - Very fast on large general purpose CPUs (modern PCs)
  - Not very easy to implement in small hardware
- Advanced Encryption Standard (AES)
  - Selected by NIST as replacement for DES in 2001
  - Uses the Rijndael algorithm
  - Keys of 128, 192 or 256 bits

Private Key Authentication

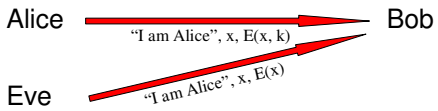
- Alice wants to talk to Bob
  - Needs to convince him of her identity
  - Both have private key k
- Naive scheme



- Vulnerability?

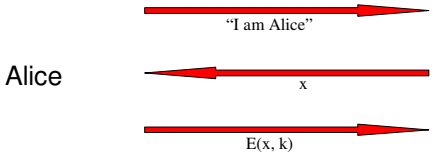
Replay Attack

- Eve can listen in and impersonate Alice later



Preventing Replay Attacks

- Bob can issue a challenge phrase to Alice



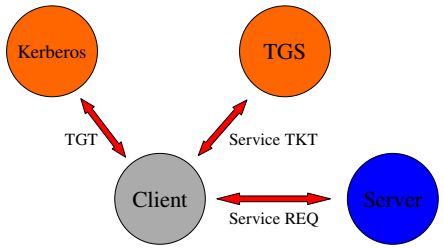
Key Distribution

- Have network with n entities
- Add one more
  - Must generate n new keys
  - Each other entity must securely get its new key
  - Big headache managing n^2 keys!
- One solution: use a central keyserver
  - Needs n secret keys between entities and keyserver
  - Generates session keys as needed
  - Downsides
    - Only scales to single organization level
    - Single point of failure

Kerberos

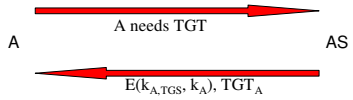
- Trivia
  - Developed in 80's by MIT's Project Athena
  - Used on all Andrew machines
  - Mythic three-headed dog guarding the entrance to Hades
- Uses DES, 3DES
- Key Distribution Center (KDC)
  - Central keyserver for a Kerberos domain
  - Authentication Service (AS)
    - Database of all master keys for the domain
    - Users' master keys are derived from their passwords
    - Generates ticket-granting tickets (TGTs)
  - Ticket Granting Service (TGS)
    - Generates tickets for communication between principals
  - "slaves" (read only mirrors) add reliability
  - "cross-realm" keys obtain tickets in others Kerberos domains

Kerberos Authentication Steps

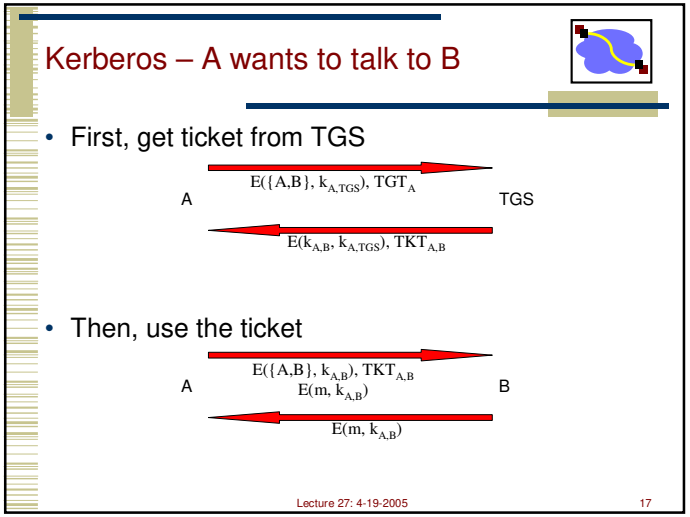


Kerberos Tickets

- What is a ticket?
  - Owner (Instance and Address)
  - A key for a pair of principles
  - A lifetime (usually ~1 day) of the key
    - Clocks in a Kerberos domain must be roughly synchronized
  - Contains all state (KDC is stateless)
  - Encrypted for server
- Ticket-granting-ticket (TGT)
  - Obtained at beginning of session
  - Encrypted with secret KDC key



Kerberos – A wants to talk to B



- First, get ticket from TGS

- Then, use the ticket

Using Kerberos

- kinit
  - Get your TGT
  - Creates file, usually stored in /tmp
- klist
  - View your current Kerberos tickets

```
unix41:~ebardsle> klist
Credentials cache: FILE:/ticket/krb5cc_61189_9FTlN6
        Principal: ebardsle@ANDREW.CMU.EDU

  Issued           Expires          Principal
Apr 18 19:40:50  Apr 19 20:40:49  krbtgt/ANDREW.CMU.EDU@ANDREW.CMU.EDU
Apr 18 19:40:50  Apr 19 20:40:49  afs@ANDREW.CMU.EDU
Apr 18 19:40:51  Apr 19 20:40:49  imap/cyrus.andrew.cmu.edu@ANDREW.CMU.EDU
```

- kdestory
  - End session, destroy all tickets
- kpasswd
  - Changes your master key stored by the AS
- "Kerberized" applications
  - kftp, ktelnet, ssh, zephyr, etc
  - afslog uses Kerberos tickets to get AFS token

Diffie-Hellman Key Agreement

- History
  - Developed by Whitfield Diffie, Martin Hellman
  - Published in 1976 paper "New Directions in Cryptography"
- Allows negotiation of secret key over insecure network
- Algorithm
  - Public parameters
    - Prime p
    - Generator g < p with property: $\forall n: 1 \le n \le p-1, \exists k: n = g^k \bmod p$
  - Alice chooses random secret a, sends Bob g^a
  - Bob chooses random secret b, sends Alice g^b
  - Alice computes $(g^b)^a$, Bob computes $(g^a)^b$ – this is the key
  - Difficult for eavesdropper Eve to compute g^{ab}

Diffie-Hellman Weakness

- Man-in-the-Middle attack
  - Assume Eve can intercept and modify packets
  - Eve intercepts g^a and g^b, then sends Alice and Bob g^c
  - Now Alice uses g^{ac}, Bob uses g^{bc}, and Eve knows both
- Defense requires mutual authentication
  - Back to key distribution problem

Public Key Cryptosystems

- Keys P, S
  - P: public, freely distributed
  - S: secret, known only to one entity
- Properties
  - x = D(E(x,S), P)
  - x = D(E(x,P), S)
  - Given x, hard to determine S(x)
  - Given P(x), hard to determine x

Using Public Key Systems

- Encryption – Bob sends to Alice
  - Bob generates and sends $m_c = E(m_p, P_A)$
  - Only Alice is able to decrypt $m_p = D(m_c, S_A)$
- Authentication – Alice proves her identity
  - Bob generates and sends challenge x
  - Alice response $s = E(x, S_A)$
  - Bob checks: $D(s, P_A) = x$
- Weakness – key distribution (again)
  - If Bob gets unauthentic P_A, he can be easily attacked
  - Solutions?

SSL/TLS

- History
  - Standard libraries and protocols for encryption and authentication
  - SSL originally developed by Netscape
    - SSL v3 draft released in 1996
  - TLS formalized in RFC2246 (1999)
- Uses
  - HTTPS, IMAP, SMTP, etc
- Issues
  - Proxies?

RSA

- Rivest, Shavir, Adleman, MIT, 1977
- Message domain
  - For large primes p, q, n = pq
    - p and q are actually strong pseudo-prime numbers generated using the Miller-Rabin primality testing algorithm
  - Messages computed over $\mathbb{Z}_n$
- Keys
  - Public key {e, n}
    - e relatively prime to (p-1)(q-1)
    - $P(x) = x^e \bmod n$
  - Private key {d, n}
    - $d = e^{-1} \bmod (p-1)(q-1)$ $(d*e = 1 \bmod (p-1)(q-1))$
    - $S(x) = x^d \bmod n$
- Strength
  - Finding d given e and n equivalent to finding p and q (factoring n)

Cryptographic Hash Functions

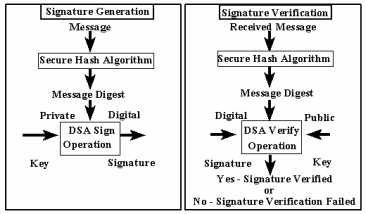
- Given arbitrary length m, compute constant length digest d = h(m)
- Desirable properties
  - h(m) easy to compute given m
  - One-way: given h(m), hard to find m
  - Weakly collision free: given h(m) and m, hard to find m' s.t. h(m) = h(m')
  - Strongly collision free: hard to find any x, y s.t. h(x) = h(y)
- Example use: password database, file distribution
- Common algorithms: MD5, SHA

Comparative Performances

- According to Peterson and Davie
- MD5: 600 Mbps
- DES: 100 Mbps
- RSA: 0.1 Mbps

Digital Signatures

- Alice wants to convince others that she wrote message m
  - Computes digest d = h(m) with secure hash
  - Signature s = $S_A(d)$
- Digital Signature Standard (DSS)

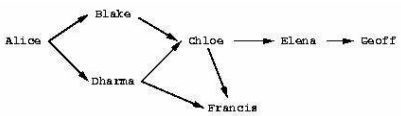


Authentication Chains

- How do you trust an unknown entity?
- Trust hierarchies
  - Certificates issued by Certificate Authorities (CAs)
    - Certificates are signed by only one CA
    - Trees are usually shallow and broad
    - Clients only need a small number of root CAs
      - Roots don't change frequently
      - Can be distributed with OS, browser
    - Example root CAs
      - VeriSign
      - Thwarte
      - CMU (for WebISO)
    - Problem
      - Root CAs have a lot of power
      - Initial distribution of root CA certificates
  - X.509
    - Certificate format standard
    - Used for SHTTP, S/MIME, others
    - Global namespace: Distinguished Names (DNs)
      - Not very tightly specified – usually includes an email address or domain name

Webs of Trust

- Anyone can generate keys
- Anyone can sign others' keys
- Trust relationships form a digraph
- Users decide how much they trust the signatures



Pretty Good Privacy (PGP)

- History
  - Written in early 1990s by Phil Zimmermann
  - Primary motivation is email security
  - Controversial for a while because it was too strong
    - Distributed from Europe
  - Now the OpenPGP protocol is an IETF standard (RFC 2440)
  - Many implementations, including the GNU Privacy Guard (GPG)
- Uses
  - Message integrity and source authentication
    - Makes message digest, signs with public key cryptosystem
    - Webs of trust
  - Message body encryption
    - Private key encryption for speed
    - Public key to encrypt the message's private key

Secure Shell (SSH)

- Negotiates use of many different algorithms
- Encryption
- Server-to-client authentication
  - Protects against man-in-the-middle
  - Uses public key cryptosystems
  - Keys distributed informally
    - kept in ~/.ssh/known_hosts
  - Signatures not used for trust relations
- Client-to-server authentication
  - Can use many different methods
  - Password hash
  - Public key
  - Kerberos tickets

IPsec

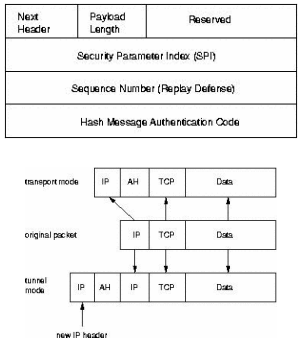
- Protection at the network layer
  - Applications do not have to be modified to get security
- Actually a suite of protocols
  - IP Authentication Header (AH)
    - Uses secure hash and symmetric key to authenticate datagram payload
  - IP Encapsulating Security Payload (ESP)
    - Encrypts datagram payload with symmetric key
  - Internet Key Exchange (IKE)
    - Does authentication and negotiates private keys

IPsec Security Associations

- Defines security for a single connection
  - Matches data sent from IP address A to IP address B
  - Uses a Security Parameter Index (SPI) as an identifier
  - Specifies encryption algorithms
  - Contains private keys for each algorithm
- Security Policy Database (SPD)
  - Specifies policies for traffic (discard, use IPsec, don't use IPsec)
- Security Association Database (SAD)
  - Contains all SAs currently used by the node
  - Can be managed by hand or with IKE

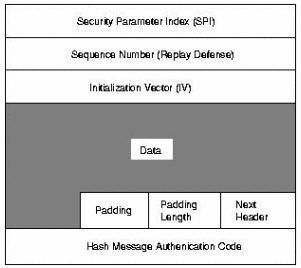
AH – Authentication Header

- Authenticates message contents
- Transport mode
  - Hashes and signs IP payload (TCP segment or UDP datagram)
  - AH goes between IP and TCP/UDP header
- Tunnel mode
  - Hashes and signs entire IP packet
  - Creates new IP header
  - AH between original and new IP headers



ESP – Encapsulated Security Payload

- Encrypts payload
- Authentication trailer optional
- Has transport and tunnel modes as well



IKE – Internet Key Exchange

- Security associations are by IP address
- What if you address changes?
  - Traveler with laptop wants to join a company's VPN
- IKE can authenticate endpoints and automatically setup security associations
  - Can use public key infrastructure (X.509) to authenticate endpoint identity
  - Can also use pre-shared private keys

Works Cited

- http://www.psc.edu/~jheffner/talks/sec_lecture.pdf
- http://en.wikipedia.org/wiki/One-time_pad
- http://www.iusmentis.com/technology/encryption/des/
- http://en.wikipedia.org/wiki/3DES
- http://en.wikipedia.org/wiki/AES
- http://en.wikipedia.org/wiki/MD5