Secure Socket Layer
Secure Socket Layer

- Introduction
- Overview of SSL
- What SSL is Useful For
Introduction

• Secure Socket Layer (SSL)

Industry-standard method for protecting web communications.

- Data encryption
- Server authentication
- Message integrity
- Optional client authentication
TCP/IP

- Designed by DOD during Cold War
- Robust
  - Multiple paths to any destination
  - Packets can arrive out of order with no problem
  - Automatically retry if data doesn’t reach destination
  - Automatic recovery from node or line failure
- Data is routed through many nodes
The Problem

- Any intermediary node could:
  - Eavesdrop
    - Intercept sensitive information
  - Tamper
    - Alter information
    - False messages
  - Impersonate
    - Pretend to be destination
The Solution

- A system that:
  - Encrypts data
    - A variety of encryption methods are available
  - Checks message integrity
    - Two hashing methods are used
  - Authenticates participants
  - https
Versions

• **SSL 1.0**
  - Limited to 40-bit keys (legal restriction)

• **SSL 2.0**
  - 168-bit keys (Triple DES)

• **SSL 3.0**
  - FORTEZZA support

• **TLS 1.0** (Transport Layer Security)
  - Based on SSL 3.0 (internal version number is 3.1)
  - Non-interoperable with SSL, only minor differences
Overview of SSL

• **SSL record protocol**
  - Format to transmit data
  - Used during handshake phase and SSL session

• **SSL handshake protocol**
  - Authenticate server to client
  - Negotiate cryptographic algorithms
  - Optionally authenticate client to server
  - Public-key encryption to generate shared secrets
  - Establish encrypted SSL connection
Hashes

- Used in conjunction with cipher keys
  - Cipher key encrypts/decrypts data
  - Hash comparison ensures message integrity

- MD5
  - 128-bit hash
  - Special pairs of messages with same hash

- SHA-1
  - 160-bit hash
  - Believed to be cryptographically secure
Key-Exchange Algorithms

- Algorithm determines keys used

- Algorithms
  - RSA — Rivest, Shamir, Adleman
  - KEA — Key Exchange Algorithm
  - DH — Diffie-Hellman
  - KRB5 — Kerberos 5
Ciphers

- Cryptographic algorithms
- SSL supports several ciphers
- Ciphers can be enabled/disabled
  - Part of handshake is determining which to use
  - Server admin can disable less-secure methods
- Prevent third-party interception
- Check message integrity
Cipher Keys

• Pseudo-random number
• Used during encryption and decryption
• Key length one component of strength
  - Longer: more possible keys, more work to guess key
• Symmetric Key
  - Same key to encrypt and decrypt messages
• Public Key
  - One key to encrypt, a different one to decrypt
Key Length

- **40-bit Keys**
  - 1 Trillion \((10^{12})\) keys
  - Breakable in a week with average PC (1997)
  - Legal limit for exportable suites until 1999

- **256-bit Keys**
  - 116 Quattuorvigintillion \((10^{75})\) keys
  - Hundreds of trillions of years with same computer
Cipher Suites

- **AES**
  - 128, 192, or 256-bit keys
  - Supersedes DES
  - No known, practical attacks

- **DES and Triple-DES (3DES)**
  - 56-bit (DES) or 168-bit (3DES) keys
  - DES Vulnerable to brute force attacks

- **IDEA**
  - 128-bit keys
  - No known, good attacks
Cipher Suites

- **RC2**
  - Variable key length (40 to 128-bit)
  - Vulnerable to related-key attack

- **RC4**
  - Variable key length (40 to 256-bit)
  - Statistically, first few bytes are non-random

- **SKIPJACK**
  - 80-bit keys
  - Classified until 1998
  - Vulnerable to impossible differential cryptanalysis
Certificates

- Used for Identity Verification
  - Public Key
  - Serial Number
  - Validity Period (cert. valid from ___ to ___)
  - Distinguishing Name (DN)
  - Issuer’s DN
  - Issuer’s Digital Signature
SSL Handshaking

• One of SSL’s strength is extensibility
  - Range of hash/key exchange/cipher combinations
• Client and Server need to agree on one
• Server needs to prove identity
• Client might need to prove identity
Client initiates contact by sending:
- SSL version (determines available cipher suites)
- Cipher settings (further limits cipher suites)
- Randomly generated data
- Any other information that might be needed
- Request for server authentication
Server Responds

- Server responds to request with:
  - Server’s SSL version
  - Server’s cipher settings
  - Another piece of randomly generated data
  - Any other information needed
  - Server’s Certificate (used to authenticate server)
  - Server may request client authentication
Server Authentication

- Is today’s date within the validity period?
- Is the Issuer trusted?
  - List of trusted Certificate Authorities (CA)
  - Issuer’s DN
  - Issuer’s public key
  - Issuer’s digital signature
- CA’s public key validate digital sig.?
- Does server’s DN match cert.’s DN?
Trusted CA List

- Clients keep list of trusted CAs
  - Issuer’s DN
  - Issuer’s public key
  - Issuer’s digital signature

- If Issuer’s DN in list all is good

- If not, check certificate chain
  - CA Hierarchies
CA Hierarchy

University

- Fine Arts
  - Staff
- Science
  - Labs
- Library
  - Networking
CA Hierarchies

- Labs Group not in list of trusted CAs
  - Labs Group is subordinate to Library, check Library

Diagram:
- University
  - Fine Arts
  - Science
  - Library
    - Staff
    - Unknown Labs CA
    - Networking
CA Hierarchies

- Library is also not in list of trusted CAs
  - Library is subordinate to University, check University
CA Hierarchies

• University is in list of trusted CAs
  - Original Issuer (Labs Group) checks out

The University of Utah
Student Computing Labs SCL
Authentication

• Everything checks out
  - Server is who it says it is, continue

• Something’s wrong
  - Warn the user that authenticated and encrypted connection cannot be established
Premaster Secret

- Client generates Premaster Secret
  - 2 bytes is the SSL version, other 46 are random
- Encrypt P.S. with server’s public key
- Send encrypted P.S. to server
- If client authentication was requested
  - Sign a hash of all SSL messages so far
  - Send signed data and client certificate with P.S.
Client Authentication

- Optional
- Does client’s public key validate sig.?
- Is today’s date within validity period?
- Is issuing CA trusted?
- Does CA’s public key validate sig.?
- Is user’s certificate listed in LDAP entry?  
  - Optional
- Is client authorized for the service?  
  - Access Control Lists (ACLs)
Master Secret

- Premaster Secret
  - Server decrypts with private key

- Server & Client generate Master Secret
  - Same P.S.
  - Same steps
  - Independent of each other
Session Keys

- Server & Client generate session keys
  - Use master secret
  - Symmetric keys to encrypt and decrypt data

- Why Symmetric cryptography?
  - Faster than Public Key cryptography
  - Since Server & Client don’t transmit actual key it is secure enough
Finishing Handshake

- **Client done**
  - Message that all future messages will be encrypted
  - Sends encrypted message that handshake is done

- **Server done**
  - Message that all future messages will be encrypted
  - Sends encrypted message that handshake is done

- **SSL Session begins**
  - Session keys used to encrypt and decrypt data
  - Session keys also used to validate messages
Uses of SSL/TLS

- eCommerce
  - Banking
  - Online stores

- secure mail (U.CC)

- secure news

- radmind
  - DHCP
  - Three levels of security (0, 1, 2)
Campus Resources

• University Internal CA
  - Managed by ISO

• Uses
  - Available for servers/applications
  - Radius
  - Time
  - VPN
  - SSL Web Servers
  - No certificates for individuals at this time
Campus Certificate

- Public certificate
  - ITAC web site (available in handout)
  - Adds U to list of trusted CAs

- Generating a certificate
  - email ca@utah.edu
  - Contact information
  - Application to use the certificate
  - Either FQDN of host or a certificate request file
Best Practices

• Hosts must meet ISO requirements
  - Vulnerability scans run before cert is issued
  - Periodic scans afterwards

• Currently no best practice document
  - ISO is accepting recommendations

• Do not use PEAP
  - Plain text password storage
Issues

• Self-Signed Certificates and Mac OS X
  - Does not automatically accept self-signed certificates
  - Security feature
  - Manually install certificate to KeyChain
  - Requires that cert be available for download

• To Add a Self-Signed Certificate
  - Download
  - Make sure the cert names ends in “.cer” or “.dem”
  - Double-Click on the certificate
  - Add to Keychain