IPSec and SSL Virtual Private Networks

ITU/APNIC/MICT IPv6 Security Workshop
23rd – 27th May 2016
Bangkok
Acknowledgment

- Content sourced from
  - Merike Kaeo of Double Shot Security
  - Contact: merike@doubleshotsecurity.com
Virtual Private Networks

- Creates a secure tunnel over a public network
- Any VPN is not automagically secure
  - You need to add security functionality to create secure VPNs
  - That means using firewalls for access control
  - And probably IPsec or SSL/TLS for confidentiality and data origin authentication
VPN Protocols

- **IPsec (Internet Protocol Security)**
  - Open standard for VPN implementation
  - Operates on the network layer
Other VPN Implementations

- **MPLS VPN**
  - Used for large and small enterprises
  - Pseudowire, VPLS, VPRN

- **GRE Tunnel**
  - Packet encapsulation protocol developed by Cisco
  - Not encrypted
  - Implemented with IPsec

- **L2TP IPsec**
  - Uses L2TP protocol
  - Usually implemented along with IPsec
  - IPsec provides the secure channel, while L2TP provides the tunnel
What is IPSec?

- IETF standard that enables encrypted communication between peers:
  - Consists of open standards for securing private communications
  - Network layer encryption ensuring data confidentiality, integrity, and authentication
  - Scales from small to very large networks
What Does IPsec Provide?

- Confidentiality....many algorithms to choose from
- Data integrity and source authentication
  - Data “signed” by sender and “signature” verified by the recipient
  - Modification of data can be detected by signature “verification”
  - Because “signature” based on a shared secret, it gives source authentication
- Anti-replay protection
  - Optional: the sender must provide it but the recipient may ignore
- Key Management
  - IKE – session negotiation and establishment
  - Sessions are rekeyed or deleted automatically
  - Secret keys are securely established and authenticated
  - Remote peer is authenticated through varying options
Different Layers of Encryption

- Application Layer – SSL, PGP, SSH, HTTPS
- Network Layer - IPsec
  - Source
  - Link Layer Encryption
  - Destination
Relevant Standard(s)

- **IETF specific**
  - rfc2409: IKEv1
  - rfc4301: IPsec Architecture (updated)
  - rfc4303: IPsec ESP (updated)
  - rfc4306: IKEv2
  - rfc4718: IKEv2 Clarifications
  - rfc4945: IPsec PKI Profile

- **IPv6 and IPsec**
  - rfc4294: IPv6 Node Requirements
  - rfc4552: Authentication/Confidentiality for OSPFv3
  - rfc4877: Mobile IPv6 Using IPsec (updated)
  - rfc4891: Using IPsec to secure IPv6-in-IPv4 Tunnels
IPsec Modes

- **Tunnel Mode**
  - Entire IP packet is encrypted and becomes the data component of a new (and larger) IP packet.
  - Frequently used in an IPsec site-to-site VPN

- **Transport Mode**
  - IPsec header is inserted into the IP packet
  - No new packet is created
  - Works well in networks where increasing a packet’s size could cause an issue
  - Frequently used for remote-access VPNs
Tunnel vs. Transport Mode IPsec

Without IPsec

Transport Mode IPsec

Tunnel Mode IPsec
Transport vs Tunnel Mode

**Transport Mode:** End systems are the initiator and recipient of protected traffic

**Tunnel Mode:** Gateways act on behalf of hosts to protect traffic
IPsec Components

- AH (Authentication Header)
  - Authentication is applied to the entire packet, with the mutable fields in the IP header zeroed out
  - If both ESP and AH are applied to a packet, AH follows ESP
  - Standard requires HMAC-MD5-96 and HMAC-SHA1-96....older implementations also support keyed MD5
IPsec Components

- ESP (Encapsulating Security Payload)
  - Must encrypt and/or authenticate in each packet
  - Encryption occurs before authentication
  - Authentication is applied to data in the IPsec header as well as the data contained as payload
  - Standard requires DES 56-bit CBC and Triple DES. Can also use RC5, IDEA, Blowfish, CAST, RC4, NULL

- IKE (Internet Key Exchange)
  - Automated SA (Security Association) creation and key management
IPsec Architecture

- **AH**: Authentication Header
- **ESP**: Encapsulating Security Payload
- **IKE**: The Internet Key Exchange

**IPsec Security Policy**
Security Associations (SA)

- A collection of parameters required to establish a secure session
- Uniquely identified by three parameters consisting of:
  - Security Parameter Index (SPI)
  - IP destination address
  - Security protocol (AH or ESP) identifier
- An SA is unidirectional
  - Two SAs required for a bidirectional communication
- A single SA can be used for AH or ESP, but not both
  - must create two (or more) SAs for each direction if using both AH and ESP
Authentication Header (AH)

- Provides source authentication and data integrity
  - Protection against source spoofing and replay attacks
- Authentication is applied to the entire packet, with the mutable fields in the IP header zeroed out
- If both AH and ESP are applied to a packet, AH follows ESP
- Operates on top of IP using protocol 51
- In IPv4, AH protects the payload and all header fields except mutable fields and IP options (such as IPsec option)
Encapsulating Security Payload (ESP)

- Uses IP protocol 50
- Provides all that is offered by AH, plus data confidentiality
  - It uses symmetric key encryption
- Must encrypt and/or authenticate in each packet
  - Encryption occurs before authentication
- Authentication is applied to data in the IPsec header as well as the data contained as payload
Internet Key Exchange (IKE)

- “An IPsec component used for performing mutual authentication and establishing and maintaining Security Associations.” (RFC 5996)
- Typically used for establishing IPsec sessions
- A key exchange mechanism
- Five variations of an IKE negotiation:
  - Two modes (aggressive and main modes)
  - Three authentication methods (pre-shared, public key encryption, and public key signature)
- Uses UDP port 500
## IKE Modes

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main Mode</strong></td>
<td>1. Three exchanges of information between IPsec peers.</td>
</tr>
<tr>
<td></td>
<td>2. Initiator sends one or more proposals to the other peer (responder), responder selects a proposal</td>
</tr>
<tr>
<td></td>
<td>3. Diffie-Hellman (DH) key exchange</td>
</tr>
<tr>
<td></td>
<td>4. Establish ISAKMP session</td>
</tr>
<tr>
<td><strong>Aggressive Mode</strong></td>
<td>1. Achieves same result as main mode using only 3 packets</td>
</tr>
<tr>
<td></td>
<td>2. First packet sent by initiator containing all info to establish SA</td>
</tr>
<tr>
<td></td>
<td>3. Second packet by responder with all security parameters selected</td>
</tr>
<tr>
<td></td>
<td>4. Third packet finalizes authentication of the ISAKMP session</td>
</tr>
<tr>
<td><strong>Quick Mode</strong></td>
<td>1. Negotiates the parameters for the IPsec session.</td>
</tr>
<tr>
<td></td>
<td>2. Entire negotiation occurs within the protection of ISAKMP session</td>
</tr>
</tbody>
</table>
Internet Key Exchange (IKE)

- **Phase I**
  - Establish a secure channel (ISAKMP SA)
  - Using either main mode or aggressive mode
  - Authenticate computer identity using certificates or pre-shared secret

- **Phase II**
  - Establishes a secure channel between computers intended for the transmission of data (IPsec SA)
  - Using quick mode
Traffic which needs to be protected is recognized as requiring IPsec protection.

IPsec with IKE

1. Traffic which needs to be protected is recognized as requiring IPsec protection.
2. Peers Authenticate using:
   - Pre-shared key
   - Digital Certificate
3. IKE Phase 2
4. IKE Phase 1
   - Secure communication channel
   - IPsec Tunnel
   - Secured traffic exchange
   - Secured Communications
IPsec IKE Phase 1 Uses DH Exchange

- First public key algorithm (1976)
- Diffie-Hellman is a key establishment algorithm
  - Two parties in a DF exchange can generate a shared secret
  - There can even be N-party DF changes where N peers can all establish the same secret key
- Diffie-Hellman can be done over an insecure channel
- IKE authenticates a Diffie-Hellman exchange
  - Pre-shared secret
  - Nonce (RSA signature)
  - Digital signature
IKE Phase 1 Main Mode

1. Negotiate IKE Policy
   - IKE Message 1 (SA proposal)
   - IKE Message 2 (accepted SA)

2. Authenticated DH Exchange
   - IKE Message 3 (DH public value, nonce)
   - IKE Message 4 (DH public value, nonce)

3. Compute DH shared secret and derive keying material

4. Protect IKE Peer Identity
   - IKE Message 5 (Authentication material, ID)
   - IKE Message 6 (Authentication material, ID) (Encrypted)
IKE Phase 2 Quick Mode

1. Message 1 (authentication/keying material and SA proposal)
2. Validate message 1
3. Message 2 (authentication/keying material and accepted SA)
4. Validate message 2
5. Message 3 (hash for proof of integrity/authentication)
6. Validate message 3
7. Compute keying material
IKE v2: Replacement for Current IKE Specification

- Feature Preservation
  - Most features and characteristics of baseline IKE v1 protocol are being preserved in v2

- Compilation of Features and Extensions
  - Quite a few features that were added on top of the baseline IKE protocol functionality in v1 are being reconciled into the mainline v2 framework

- Some New Features
IKE v2: What Is Not Changing

- Features in v1 that have been debated but are ultimately being preserved in v2
  - Most payloads reused
  - Use of nonces to ensure uniqueness of keys

- v1 extensions and enhancements being merged into mainline v2 specification
  - Use of a ‘configuration payload’ similar to MODECFG for address assignment
  - ‘X-auth’ type functionality retained through EAP
  - Use of NAT Discovery and NAT Traversal techniques
IKE v2: What Is Changing

- Significant Changes Being to the Baseline Functionality of IKE
  - EAP adopted as the method to provide legacy authentication integration with IKE
  - Public signature keys and pre-shared keys, the only methods of IKE authentication
  - Use of ‘stateless cookie’ to avoid certain types of DOS attacks on IKE
  - Continuous phase of negotiation
How Does IKE v2 Work?

IKE_SA_INIT
(Two Messages)
IKE_SA Authentication
Parameters Negotiated
IKE_AUTH
(Two Messages)
IKE Authentication Occurs
and One CHILD_SA Created
CREATE_CHILD_SA
(Two Messages)
Second CHILD_SA Created
Protected Data
IPv4 IPsec AH

IPv4 AH

**Transport Mode:**

*Before applying AH:*

Original IP Header | TCP/UDP | Data

*After applying AH:*

Original IP Header | AH Header | TCP/UDP | Data

Authenticated except for mutable fields in IP header

**Mutable Fields:**
- ToS
- TTL
- Hdr Checksum
- Offset
- Flags

IPv4 AH

**Tunnel Mode:**

*Before applying AH:*

Original IP Header | TCP/UDP | Data

*After applying AH:*

New IP Header | AH Header | Original IP Header | Data

Authenticated except for mutable fields in new IP header

**Mutable Fields:**
- ToS
- TTL
- Hdr Checksum
- Offset
- Flags
IPv4 IPsec ESP

IPv4 ESP
Transport Mode:

Before applying ESP:

Original IP Header | TCP/UDP | Data

After applying ESP:

Original IP Header | ESP Header | TCP/UDP | Data | ESP Trailer | ESP Auth

Authenticated

Encrypted

IPv4 ESP
Tunnel Mode:

Before applying ESP:

Original IP Header | TCP/UDP | Data

After applying ESP:

New IP Header | ESP Header | Original IP Header | TCP/UDP | Data | ESP Trailer | ESP Auth

Authenticated

Encrypted
ESP Header Format

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Parameter Index (SPI)</td>
</tr>
<tr>
<td>Sequence Number</td>
</tr>
<tr>
<td>Initialization Vector (IV)</td>
</tr>
<tr>
<td>Payload Data (Variable)</td>
</tr>
<tr>
<td>Padding (0-255 bytes)</td>
</tr>
<tr>
<td>Padding Length</td>
</tr>
<tr>
<td>Next Header</td>
</tr>
<tr>
<td>Authentication Data (ICV)</td>
</tr>
</tbody>
</table>

SPI: Arbitrary 32-bit number that specifies SA to the receiving device
Seq #: Start at 1 and must never repeat; receiver may choose to ignore
IV: Used to initialize CBC mode of an encryption algorithm
Payload Data: Encrypted IP header, TCP or UDP header and data
Padding: Used for encryption algorithms which operate in CBC mode
Padding Length: Number of bytes added to the data stream (may be 0)
Next Header: The type of protocol from the original header which appears in the encrypted part of the packet
Auth Data: ICV is a digital signature over the packet and it varies in length depending on the algorithm used (SHA-1, MD5)
Considerations For Using IPsec

- Security Services
  - Data origin authentication
  - Data integrity
  - Replay protection
  - Confidentiality

- Size of network

- How trusted are end hosts – can a-priori communication policies be created?

- Vendor support

- What other mechanisms can accomplish similar attack risk mitigation
Non-Vendor Specific Deployment

Issues

- Historical Perception
  - Configuration nightmare
  - Not interoperable

- Performance Perception
  - Need empirical data
  - Where is the real performance hit?

- Standards Need Cohesion
Vendor Specific Deployment Issues

- Lack of interoperable defaults
  - A default does NOT mandate a specific security policy
  - Defaults can be modified by end users

- Configuration complexity
  - Too many knobs
  - Vendor-specific terminology

- Good News: IPv6 support in most current implementations
IPsec Concerns

- Are enough people aware that IKEv2 is not backwards compatible with IKEv1?
  - IKEv1 is used in most IPsec implementations
  - Will IKEv2 implementations first try IKEv2 and then revert to IKEv1?

- Is IPsec implemented for IPv6?
  - Some implementations ship IPv6 capable devices without IPsec capability and host requirements is changed from MUST to SHOULD implement

- OSPFv3
  - All vendors ‘IF’ they implement IPsec used AH
  - Latest standard to describe how to use IPsec says MUST use ESP w/Null encryption and MAY use AH
IPsec Concerns (cont)

- What is transport mode interoperability status?
  - Will end user authentication be interoperable?

- PKI Issues
  - Which certificates do you trust?
  - How does IKEv1 and/or IKEv2 handle proposals with certificates?
  - Should common trusted roots be shipped by default?
  - Who is following and implementing pki4ipsec-ikecert-profile (rfc4945)

- Have mobility scenarios been tested?
  - Mobility standards rely heavily on IKEv2

- ESP – how to determine if ESP-Null vs Encrypted
## Default Issues

<table>
<thead>
<tr>
<th>Vendor A</th>
<th>Vendor B</th>
<th>Vendor C</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IKE Phase 1</strong></td>
<td><strong>IKE Phase 1</strong></td>
<td><strong>IKE Phase 1</strong></td>
</tr>
<tr>
<td>SHA1</td>
<td>MD5</td>
<td>SHA1</td>
</tr>
<tr>
<td>RSA-SIG</td>
<td>Pre-Share Key</td>
<td>Pre-Share Key</td>
</tr>
<tr>
<td>Group 1</td>
<td>Group 5</td>
<td>Group 2</td>
</tr>
<tr>
<td>Lifetime 86400 Sec</td>
<td>Lifetime 86400 Sec</td>
<td>Lifetime 86400 Sec</td>
</tr>
<tr>
<td>Main Mode</td>
<td>Main Mode</td>
<td>Aggressive Mode</td>
</tr>
<tr>
<td><strong>IKE Phase 2</strong></td>
<td><strong>IKE Phase 2</strong></td>
<td><strong>IKE Phase 2</strong></td>
</tr>
<tr>
<td>PFS</td>
<td>PFS</td>
<td>PFS</td>
</tr>
<tr>
<td>Group 1</td>
<td>Group 5</td>
<td>Group 2</td>
</tr>
</tbody>
</table>
Terminology Issues

- IKE Phase 1
  - IKE Phase 1 SA
  - IKE SA
  - ISAKMP SA
  - Main Mode

- DH Key Length
  - DH Group
  - Modp #
  - Group #

- IKE Phase 2
  - IKE Phase 2 SA
  - IPsec SA
  - Quick Mode

Configuration complexity increased with vendor specific configuration terms
Syslog server 2001:DB8:6665:AF75::3D authenticate esp-null sha1 pre-share ‘secret4syslog’
TFTP server 2001:DB8:6665:AF75::3D authenticate esp-null aes128 pre-share ‘secret4tftp’
BGP peer 2001:DB8:8888:BAD::66 authenticate esp-null aes128 pre-share ‘secret4AS#XXX’
Interoperable Defaults For SAs

- Security Association: groups elements of a conversation together
  - ESP encryption algorithm and key(s)
  - Cryptographic synchronization
  - SA lifetime
  - SA source address
  - Mode (transport or tunnel)

How Do We Communicate Securely?

- Do we want integrity protection of data?
- Do we want to keep data confidential?
- Which algorithms do we use?
- What are the key lengths?
- When do we want to create new keys?
- Are we providing security end-to-end?
Pretty Good IPsec Policy

- **IKE Phase 1** (aka ISAKMP SA or IKE SA or Main Mode)
  - 3DES (AES-192 if both ends support it)
  - Lifetime (8 hours = 480 min = 28800 sec)
  - SHA-2 (256 bit keys)
  - DH Group 14 (aka MODP# 14)

- **IKE Phase 2** (aka IPsec SA or Quick Mode)
  - 3DES (AES-192 if both ends support it)
  - Lifetime (1 hour = 60 min = 3600 sec)
  - SHA-2 (256 bit keys)
  - PFS 2
  - DH Group 14 (aka MODP# 14)
Sample Router Configuration

crypto isakmp policy 1
  authentication pre-share
  encryption aes
  hash sha
  group 5
crypto isakmp key Training123 address 172.16.11.66
!crypto ipsec transform-set ESP-AES-SHA esp-aes esp-sha-hmac
!crypto map LAB-VPN 10 ipsec-isakmp
  match address 101
  set transform-set ESP-AES-SHA
  set peer 172.16.11.66
Sample Router Configuration

interface FastEthernet 0/1
  crypto map LAB-VPN
exit
!
access-list 101 permit ip 172.16.16.0 0.0.0.255 172.16.20.0 0.0.0.255
Help With Configuring IPsec

- Documentation Profiles for IPsec Interoperability
  - http://www.vpnc.org/InteropProfiles/

- Documents for Cisco IPsec configuration:

- Document for Juniper IPsec configuration:
  - http://kb.juniper.net/InfoCenter/index?page=content&id=KB10128
Capture: Telnet
### Capture: Telnet + IPsec

<table>
<thead>
<tr>
<th>176</th>
<th>67.462003</th>
<th>2001:0db8:0:0:0</th>
<th>192.168.1.100</th>
<th>108FV0</th>
<th>88 Neighbor Solicitation for 2001</th>
</tr>
</thead>
<tbody>
<tr>
<td>179</td>
<td>67.594031</td>
<td>192.168.1.1</td>
<td>192.168.1.2</td>
<td>ESP</td>
<td>134 ESP (SPI=0x7ea7f8ed)</td>
</tr>
<tr>
<td>180</td>
<td>67.601908</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>181</td>
<td>67.601910</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>182</td>
<td>67.605809</td>
<td>192.168.1.1</td>
<td>192.168.1.2</td>
<td>ESP</td>
<td>118 ESP (SPI=0x7ea7f8ed)</td>
</tr>
<tr>
<td>183</td>
<td>67.626089</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>134 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>184</td>
<td>67.626091</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>134 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>185</td>
<td>67.627695</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>166 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>186</td>
<td>67.627697</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>166 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>187</td>
<td>67.631728</td>
<td>192.168.1.1</td>
<td>192.168.1.2</td>
<td>ESP</td>
<td>118 ESP (SPI=0x7ea7f8ed)</td>
</tr>
<tr>
<td>188</td>
<td>67.632884</td>
<td>192.168.1.1</td>
<td>192.168.1.2</td>
<td>ESP</td>
<td>118 ESP (SPI=0x7ea7f8ed)</td>
</tr>
<tr>
<td>189</td>
<td>67.751716</td>
<td>192.168.1.1</td>
<td>192.168.1.2</td>
<td>ESP</td>
<td>150 ESP (SPI=0x7ea7f8ed)</td>
</tr>
<tr>
<td>190</td>
<td>67.756217</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>191</td>
<td>67.756219</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>192</td>
<td>67.766534</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>193</td>
<td>67.766536</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>194</td>
<td>67.768056</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>195</td>
<td>67.768058</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>196</td>
<td>67.769385</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>197</td>
<td>67.769387</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>198</td>
<td>67.770903</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>199</td>
<td>67.770804</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
<td>ESP</td>
<td>118 ESP (SPI=0x742f79b4)</td>
</tr>
<tr>
<td>200</td>
<td>67.772002</td>
<td>192.168.1.1</td>
<td>192.168.1.2</td>
<td>ESP</td>
<td>134 ESP (SPI=0x742f79b4)</td>
</tr>
</tbody>
</table>
SSL/TLS

- Most widely-used protocol for security
- Encrypts the segments of network connections above the Transport Layer

SSL and TLS
- SSL v3.0 specified in an I-D in 1996 (draft-freier-ssl-version3-02.txt)
- TLS v1.0 specified in RFC 2246 in 1999
- TLS v1.0 = SSL v3.1 ≈ SSL v3.0
- TLS v1.1 in 2006
- TLS v1.2 in 2008

Goals of protocol
- Secure communication between applications
- Data encryption
- Server authentication
- Message integrity
- Client authentication (optional)
Some Applications Using TLS/SSL

- Securing WWW traffic (HTTPS)
- Browsers Apache
  - Apache_mod_ssl
- DNSSEC requires SSL
- Postfix, Sendmail, SMTP
- sTelnet
- OpenSSH
- SFTP
- SSL VPNs such as OpenVPN and OpenConnect
- VoIP and SIP signaling
- EAP-TLS for WiFi

http://www.openssl.org/related/apps.html
Benefits of TLS

- Application-layer independent
  - can be implemented with any applications
  - a wide range of applications supporting it

- SSL makes use of both asymmetric and symmetric key cryptography.
  - performance reasons.
  - Only the initial "client key exchange message" is encrypted with asymmetric encryption.
  - Symmetric encryption is better in terms of performance/speed
Benefits of TLS (cont)

- Uses X.509 certificates
  - Certificates and Public Key Infrastructure
- SSL protocol layers comes on top of TCP (transport Layer), and is below application layer.
  - No network infrastructure changes are required to deploy SSL
- Each and every connection that's made, through SSL has got one session information.
  - Session can also be reused or resumed for other connections to the server
SSL/TLS Properties

- **Connection is private**
  - Encryption is used after an initial handshake to define a secret key.
  - Symmetric cryptography used for data encryption

- **Peer’s identity can be authenticated**
  - Asymmetric cryptography is used (RSA or DSS)

- **Connection is reliable**
  - Message transport includes a message integrity check using a keyed MAC.
  - Secure hash functions (such as SHA and MD5) are used for MAC computations.
SSL Protocol Building Blocks

SSL is a Combination of a Primary Record Protocol with Four ‘Client’ Protocols

- SSL Handshake Protocol
- SSL Alert Protocol
- SSL Change Cipher Spec Protocol
- Application Data Protocol

SSL Record Protocol
SSL Protocol Building Block Functions

- **SSL Handshake Protocol**: Negotiates crypto algorithms and keys.
- **SSL Alert Protocol**: Indicates error or the end of a session.
- **SSL Change Cipher Spec Protocol**: Used to signal transition to new cipher and keys generally towards the end of a handshake negotiation.
- **SSL Record Protocol**: Indicates which encryption and integrity protection is applied to the data.
SSL Record Layer

- Provides fragmentation, compression, integrity protection, and encryption for data objects exchanged between clients and servers
- Maintains a current and a pending connection state
- Upper Layer → TLSPlaintext → TLSCompressed → TLSCiphertext → (send to transport)
The SSL Handshake Process

SSL Client \rightarrow Internet \rightarrow SSL Server

1. ClientHello
   - Client initiates SSL connection / sends supported cipher suites

2. ServerHello
   - Server returns digital certificate to client and selected cipher suite
   - Certificate
   - Certificate

3. ClientKeyExchange
   - Client sends shared secret encrypted with server's public key

4. ChangeCipherSpec
   - Message encryption and integrity algorithms are negotiated

5. Finished
   - Session keys are generated

6. Secure session tunnel is established

SSL version, Random data (ClientHello.random), sessionID, cipher
suits, compression algorithm

Client computes the premaster key

SSL version, Cipher suits, Random data (ServerHello.random), sessionID
<- Application Data ->

<Diagram of the SSL Handshake Process>
SSL Client Authentication

- Client authentication (certificate based) is optional and not often used.
- Many application protocols incorporate their own client authentication mechanism such as username/password or S/Key.
- These authentication mechanisms are more secure when run over SSL.
## SSL/TLS IANA Assigned Port #s

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Defined Port Number</th>
<th>SSL/TLS Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
<td>80</td>
<td>443</td>
</tr>
<tr>
<td>NNTP</td>
<td>119</td>
<td>563</td>
</tr>
<tr>
<td>POP</td>
<td>110</td>
<td>995</td>
</tr>
<tr>
<td>FTP-Data</td>
<td>20</td>
<td>989</td>
</tr>
<tr>
<td>FTP-Control</td>
<td>21</td>
<td>990</td>
</tr>
<tr>
<td>Telnet</td>
<td>23</td>
<td>992</td>
</tr>
</tbody>
</table>
## Capture: SSL Decryption (easy)

<table>
<thead>
<tr>
<th>Sequence</th>
<th>IP Address</th>
<th>Port</th>
<th>Protocol</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 0 660037</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 36713 &gt; https [ACK] Seq=1 Ack=1 Win=32767 Len=0 TSecr=255562115 TSecr=255562115</td>
</tr>
<tr>
<td>5 0 660178</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 https:38713 [ACK] Seq=1 Ack=107 Win=32767 Len=0 TSecr=255562115 TSecr=255562115</td>
</tr>
<tr>
<td>5 0 902160</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>955 Server Hello, Certificate, Server Hello Done</td>
</tr>
<tr>
<td>6 0 662899</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>66 36713 &gt; https [ACK] Seq=106 Ack=930 Win=32767 Len=0 TSecr=255565137 TSecr=255565137</td>
</tr>
<tr>
<td>8 0 902693</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>278 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message</td>
</tr>
<tr>
<td>9 0 822270</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>141 Change Cipher Spec, Encrypted Handshake Message</td>
</tr>
<tr>
<td>10 0 822809</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>66 36713 &gt; https [ACK] Seq=318 Ack=1005 Win=32767 Len=0 TSecr=255564938 TSecr=255564938</td>
</tr>
<tr>
<td>11 0 833074</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>503 Application Data</td>
</tr>
<tr>
<td>12 0 873235</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 https:38713 [ACK] Seq=1005 Ack=755 Win=32767 Len=0 TSecr=255564989 TSecr=255564989</td>
</tr>
<tr>
<td>13 0 938485</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>103 Encrypted Handshake Message</td>
</tr>
<tr>
<td>13 0 938750</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>183 Encrypted Handshake Message</td>
</tr>
<tr>
<td>15 0 938761</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>66 https:38713 [ACK] Seq=1842 Ack=872 Win=32767 Len=0 TSecr=255565054 TSecr=255565054</td>
</tr>
<tr>
<td>16 0 939899</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>1973 Encrypted Handshake Message, Encrypted Handshake Message, Encrypted Handshake Message</td>
</tr>
<tr>
<td>17 0 940026</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>337 Encrypted Handshake Message, Change Cipher Spec, Encrypted Handshake Message</td>
</tr>
<tr>
<td>19 0 943406</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>172 Change Cipher Spec, Encrypted Handshake Message</td>
</tr>
<tr>
<td>19 0 944025</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>575 Application Data, Application Data</td>
</tr>
<tr>
<td>20 0 944854</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 36713 &gt; https [ACK] Seq=143 Ack=7845 Win=32767 Len=0 TSecr=255565060 TSecr=255565060</td>
</tr>
<tr>
<td>21 0 964424</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>471 Application Data</td>
</tr>
<tr>
<td>33 0 964256</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 https:38713 [ACK] Seq=7845 Ack=548 Win=32767 Len=0 TSecr=255565120 TSecr=255565120</td>
</tr>
</tbody>
</table>

### Using stolen key file

<table>
<thead>
<tr>
<th>Sequence</th>
<th>IP Address</th>
<th>Port</th>
<th>Protocol</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>26 2 964819</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 https:38714 [ACK] Seq=1 Ack=121 Win=32767 Len=0 TSecr=255565080 TSecr=255565080</td>
</tr>
<tr>
<td>27 2 902274</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>228 Server Hello, Change Cipher Spec, Finished</td>
</tr>
<tr>
<td>28 2 902312</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>SSLv3</td>
<td>66 38714 &gt; https [ACK] Seq=121 Ack=155 Win=32767 Len=0 TSecr=255565108 TSecr=255565108</td>
</tr>
<tr>
<td>29 2 902855</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>562 GET /icons/debian/openlogo-25.jpg HTTP/1.1</td>
</tr>
<tr>
<td>30 2 992501</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>596 HTTP/1.1 404 Not Found (text/html)</td>
</tr>
<tr>
<td>31 2 993849</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>471 GET /icons/apache_pb.png HTTP/1.1</td>
</tr>
<tr>
<td>32 2 994179</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>1828 HTTP/1.1 200 OK (PNG)</td>
</tr>
<tr>
<td>33 2 994256</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 https:38713 [ACK] Seq=7845 Ack=1548 Win=32767 Len=0 TSecr=255565128 TSecr=255565128</td>
</tr>
<tr>
<td>34 3 633250</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 38714 &gt; https [ACK] Seq=1022 Ack=2447 Win=32767 Len=0 TSecr=255565149 TSecr=255565149</td>
</tr>
<tr>
<td>35 3 501643</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>588 HTTP/1.1 404 Not Found (text/html)</td>
</tr>
<tr>
<td>36 3 507091</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>439 GET /favicon.ico HTTP/1.1</td>
</tr>
<tr>
<td>37 3 507541</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>580 HTTP/1.1 404 Not Found (text/html)</td>
</tr>
<tr>
<td>38 3 507555</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 38714 &gt; https [ACK] Seq=1395 Ack=2961 Win=32767 Len=0 TSecr=255565223 TSecr=255565223</td>
</tr>
<tr>
<td>39 3 544174</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 38713 &gt; https [ACK] Seq=1548 Ack=8367 Win=32767 Len=0 TSecr=25556567 TSecr=25556567</td>
</tr>
<tr>
<td>40 6 037808</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>511 GET /test HTTP/1.1</td>
</tr>
<tr>
<td>41 6 037932</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>TCP</td>
<td>66 https:38713 [ACK] Seq=8367 Ack=1993 Win=32767 Len=0 TSecr=255560154 TSecr=255560154</td>
</tr>
<tr>
<td>42 6 041185</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>HTTP</td>
<td>644 HTTP/1.1 301 Moved Permanently (text/html)</td>
</tr>
</tbody>
</table>
Attacks on SSL (a little harder...)

- **BEAST Attack (2011)**
  - Browser Exploit Against SSL/TLS
  - CBC vulnerability discovered in 2002
  - Fixed in TLS 1.1

- **CRIME Attack (2012)**
  - Compression Ratio Info-leak Made Easy
  - Exploit against TLS compression
  - ‘fixed’ by disabling TLS Compression

- **BREACH Attack (2013)**
  - Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext
  - Presented at BlackHat 2013 (Aug)
  - Attacks HTTP responses using HTTP Compression
Encrypted Communications

- Use encrypted communications whenever you need to keep information confidential
- Verify via network sniffer (e.g. wireshark) that your communication is indeed encrypted
- An important aspect is credential management (creating, distributing, storing, revoking, renewing)
- Understand if/when credentials are lost that you may not be able to recover the data
- Have a plan in place in case you forget your password that protects your private keys
Thank You. Questions?
IPSec and SSL Virtual Private Networks

ITU/APNIC/MICT IPv6 Security Workshop
23rd – 27th May 2016
Bangkok