Network Security: Conversation Isolation

VPN & TLS

Firewalls

Fundamental Layer 2 & 3 Problems

- IP relies on store-and-forward networking
  - Network data passes through untrusted hosts
  - Routes may be altered to pass data through malicious hosts
- Packets can be sniffed (and new forged packets injected)
- Ethernet, IP, TCP & UDP
  - All designed with no authentication or integrity mechanisms
  - No source authentication on IP packets – they might be forged
  - TCP session state can be examined or guessed ...
  - and then TCP sessions can be hijacked
  - Man-in-the-middle attacks are possible
- ARP, DHCP, DNS protocols
  - Can be spoofed to redirect traffic to malicious hosts
- Internet route advertisement protocols are not secure
  - Can redirect traffic to malicious routers/hosts

Solution: Use private networks

Connect multiple geographically-separated private subnetworks together

Virtual Private Networks

Take the concept of tunneling
... and safeguard the encapsulated data

- Add a MAC (message authentication code)
  - Ensure that outsiders don’t modify the data
- Encrypt the contents
  - Ensure that outsiders can’t read the data
IPsec
Internet Protocol Security
End-to-end solution at the IP layer
Two protocols:
• IP Authentication Header Protocol (AH)
  – Authentication & integrity of payload and header
  – Provides integrity
• Encapsulating Security Payload (ESP)
  – AH + Confidentiality of payload
  – Adds content encryption

Tunnel mode vs. transport mode
Tunnel mode VPN
– Communication between gateways: network-to-network
– Entire IP datagram is encapsulated
  • The system sends IP packets to various addresses on subnet
  • A router (tunnel endpoint) on the remote side extracts the datagram and routes it on the internal network
Transport mode VPN
– Communication between hosts
– IP header is not modified
  • The system communicates directly with only one other system
  • Note: this does not operate at the transport layer – IP datagrams can be sent to various services on the host

IPsec Authentication Header (AH)
Guarantees integrity & authenticity of IP packets
– MAC for the contents of the entire IP packet
– Computed over unchangeable IP datagram fields
  (e.g., not TTL or fragmentation fields)
Protects from:
  – Tampering
  – Forging addresses
  – Replay attacks (sequence number in MAC-protected AH)

IPsec Encapsulating Security Payload (ESP)
Encrypts entire payload
– Plus authentication of payload and IP header (everything AH does)
  (may be optionally disabled – but you don’t want to)

IPSec algorithms
Authentication
– Certificates, or pre-shared key authentication
  • Public keys in certificates (RSA or ECC) used for authenticating users
    (prove you have a private key by decrypting data encrypted with the public key in your certificate)
  • Pre-shared = configure a shared key ahead of time
Key exchange = Diffie-Hellman
  – Diffie-Hellman to exchange public keys for key generation
  – Key lifetimes determine when new keys are regenerated
  – Random key generation ensures Forward Secrecy
Confidentiality = symmetric algorithm
  – 3DES-CBC
  – AES-CBC
Integrity protection & authenticity – MACs
  – HMAC-SHA1
  – HMAC-SHA2

Transport Layer Conversation Isolation:
Transport Layer Security (TLS)
Network vs. Transport Layer

VPNs were designed to operate at the network layer
- Connect networks together
- They establish a secure communication channel that can then be shared by multiple applications
- Applications are not aware that the VPN is there

What if we want to talk to a network service, such as a web server ... but securely?
- VPNs aren’t an easy answer
- We want to do this at the transport layer – for a single application talking to a service on a socket

Transport Layer Security

Goal: provide a transport layer security protocol
After setup, applications feel like they are using TCP sockets

SSL: Secure Socket Layer
Created with HTTP in mind
- Web sessions should be secure
  - Encrypted, tamperproof, resilient to man-in-the-middle attacks
  - Mutual authentication is usually not needed
  - Client needs to identify the server but the server isn’t expected to know all clients
  - Rely on password authentication after the secure channel is set up

Goal
Provide authentication (usually one-way), privacy, & data integrity between two applications

Principles
- Data encryption
  - Use symmetric cryptography to encrypt data
  - Key exchange: keys generated uniquely at the start of each session
- Data integrity
  - Include a MAC with transmitted data to ensure message integrity
- Authentication
  - Use public key cryptography & X.509 certificates for authentication
  - Optional – can authenticate 0, 1, or both parties
- Interoperability & evolution
  - Support many different key exchange, encryption, integrity, & authentication protocols – negotiate what to use at the start of a session

TLS vs. SSL – versions

SSL evolved to TLS (Transport Layer Security)
SSL 3.0 was the last version of SSL
... and is considered insecure

We now use TLS (but is often still called SSL)
- TLS 1.0 = SSL 3.1, TLS 1.1 = SSL 3.2, TLS 1.2 = SSL 3.3
- Latest version = TLS 1.3 = SSL 3.4

- Retired versions
  - TLS 1.0/SSL 3 are not considered strong anymore and their use is not recommended
  - As of 2019, Google Chrome deprecates support for TLS 1.1

TLS Protocol

Two sub-protocols

1. Authenticate & establish keys
   - Authentication
     - Public keys (X.509 certificates and – usually – RSA cryptography)
     - Key exchange options
       - Ephemeral Diffie-Hellman keys (generated for each session)
     - Pre-shared key

2. Communicate
   - Data encryption options – symmetric cryptography
     - AES GCM, AES CBC, ARIA (GCM/CBC), ChaCha20/Poly1305, ...
   - Data integrity options – message authentication codes
     - HMAC-MD5, HMAC-SHA1, HMAC-SHA256/384, ...

TLS Protocol & Ciphers

Two sub-protocols

1. Authenticate & establish keys
   - Authentication
     - Public keys (X.509 certificates and – usually – RSA cryptography)
     - Key exchange options
       - Ephemeral Diffie-Hellman keys (generated for each session)
     - Pre-shared key

2. Communicate
   - Data encryption options – symmetric cryptography
     - AES GCM, AES CBC, ARIA (GCM/CBC), ChaCha20/Poly1305, ...
   - Data integrity options – message authentication codes
     - HMAC-MD5, HMAC-SHA1, HMAC-SHA256/384, ...

TLS Protocol

(1) Client Hello
Version & cipher information
(2) Server Hello
Server certificate
[client certificate request]
(3) Verify server certificate
(4) Client key exchange (2-3)
Send encrypted session key
[ (5) Send client certificate ]
(6) Verify client certificate
(7) Client done
(8) Server done
(9) Communicate
Symmetric encryption + HMAC
Benefits of TLS

Benefits
- Protects integrity of communications
- Protects the privacy of communications
- Validates the authenticity of the server (if you trust the CA)

Some past attacks on TLS

- Man-in-the-middle: BEAST attack in TLS 1.0
  - Attacker was able to see Initialization Vector (IV) for CBC and deduce plaintext (because of known HTML headers & cookies)
  - An IV doesn’t have to be secret – but it turned out this wasn’t a good idea
  - Attacker was able to send chosen plaintext & get it encrypted with a known IV
  - Fixed by using fresh IVs for each new 16K block
- Man-in-the-middle: crypto renegotiation
  - Attacker can renegotiate the handshake protocol during the session to disable encryption
  - Proposed fix: have client & server verify info about previous handshakes
- THC-SSL-DoS attack
  - Attacker initiates a TLS handshake & requests a renegotiation of the encryption key – repeat over & over, using up server resources

Client authentication Problem

- SSL supports mutual authentication
  - Clients can authenticate servers & servers can authenticate clients
- Client authentication is almost never used
  - Generating keys & obtaining certificates is not an easy process for users
  - Any site can request the user’s certificate
  - User will be unaware their anonymity is lost
  - Moving private keys around can be difficult
  - What about users on shared or public computers?
- We usually rely on other authentication mechanisms
  - Usually user name and password
  - But there no danger of eavesdropping since the session is encrypted
  - May use one-time passwords or two-factor authentication if worried about eavesdroppers at physical premises

Network Security Goals

- Confidentiality: sensitive data & systems not accessible
- Integrity: data not modified during transmission
- Availability: systems should remain accessible
Firewall

- Separate your local network from the Internet
  - Protect the border between trusted internal networks and the untrusted Internet

- Approaches
  - Packet filters
  - Application proxies
  - Intrusion detection / intrusion protection systems

Packet Filters

Screening router

- **Border router** (gateway router)
  - Router between the internal network(s) and external network(s)
  - Any traffic between internal & external networks passes through the border router

  Instead of just routing the packet, decide whether to route it

- **Screening router = Packet filter**
  - Allow or deny packets based on
  - Incoming & outgoing interfaces
  - Source & destination IP addresses
  - Source & destination TCP/UDP ports, ICMP command
  - Protocol (e.g., TCP, UDP, ICMP, IGMP, RSVP, etc.)

Filter chaining

An IP packet entering a router is matched against a set of rules: access control list (ACL) or chain

Each rule contains criteria and an action

- Criteria: packet screening rule
- Actions
  - **Accept** – and stop processing additional rules
  - **Drop** – discard the packet and stop processing additional rules
  - **Reject** – and send an error to the sender (ICMP Destination Unreachable)
- Also
  - **Route** – reroute packets
  - **Nat** – perform network address translation
  - **Log** – record the activity

Filter structure is vendor specific

Examples

- **Windows**
  - **Allow, Block**
    - Options such as
      - Discard all traffic except packets allowed by filters (default deny)
      - Pass all traffic except packets prohibited by filters (default allow)
  - **OpenBSD**
    - **Pass (allow), Block**
    - Linux **rules (netfilter)**
      - Chain types: **filter, route, nat**
      - Chain control
        - **Return** – stop traversing a chain
        - **Jump** – jump to another chain (goto = same but no return)

Network Ingress Filtering: incoming packets

Basic firewalling principle
No direct inbound connections external systems (Internet) to any internal host – all traffic must flow through a firewall and be inspected

- Determine which services you want to expose to the Internet
  - e.g., HTTP & HTTPS: TCP ports 80 and 443

- Create a list of services and allow only those inbound ports and protocols to the machines hosting the services.

- **Default Deny** model - by default, “deny all”
  - Anything not specifically permitted is dropped
  - May want to log denies to identify who is attempting access
Network Ingress Filtering (inbound)

- Disallow IP source address spoofing
  - Restrict forged traffic (RFC 2827)
- At the ISP
  - Filter upstream traffic - prohibit an attacker from sending traffic from forged IP addresses
  - Attacker must use a valid, reachable source address
- Disallow incoming/outgoing traffic from private, non-routable IP addresses
  - Helps with DDoS attacks such as SYN flooding from lots of invalid addresses

access-list 199 deny ip 192.168.0.0 0.0.255.255 any log
access-list 199 deny ip 224.0.0.0 0.0.0.255 any log
....
access-list 199 permit ip any any

Network Egress Filtering (outbound)

- Usually we don’t worry about outbound traffic
  - Communication from a higher security network (internal) to a lower security network (Internet) is usually fine
- Why might we want to restrict it?
  - Consider: if a web server is compromised & all outbound traffic is allowed, it can connect to an external server and download more malicious code... or launch a DoS attack on the internal network
- Also, log which servers are trying to access external addresses

Stateful Inspection – 2nd generation firewalls

- Retain state information about a stream of related packets
- Examples
  - TCP connection tracking
    - Disallow TCP data packets unless a connection is set up
  - ICMP echo-reply
    - Allow ICMP echo-reply only if a corresponding echo request was sent.
  - Related traffic
    - Identify & allow traffic that is related to a connection
    - Example: related ports in FTP
      - Client connects to server on port 21 to send commands
      - Server connects back to client on port 20 to send data

Network Design: DMZ

- Clients from the Internet:
  - Can access allowed services in the DMZ
  - Cannot access internal hosts
  - The router: Blocks impersonated packets
- Clients in the internal subnet:
  - Can access the Internet
  - Can access allowed services in the DMZ
  - May access extra services in the DMZ
Network Design: DMZ

- DMZ subnet
- Internet
- Security Appliance (screening router)
- Internal subnet

Clients in the DMZ:
- Can access Internet services only to the extent required
- Can access internal services only to the extent required

Goal: Limit possible damage if DMZ machines are compromised

Network Address Translation

- Most organizations use private IP addresses
- External traffic goes through a NAT router
  - Network Address Translation
- NAT is an implicit firewall (sort of)
  - Arbitrary hosts and services on them (ports) cannot be accessed unless
    - They are specifically mapped to a specific host/port by the administrator
    - Internal services have initiated outgoing traffic
      - Return traffic from the same address/port will be accepted

Application-Layer Filtering

Firewalls don't work well when everything is a web service

Deep packet inspection (DPI)
- Look beyond layer 3 & 4 headers
- Need to know something about application protocols & formats

Examples
- URL filtering
  - Normal source/destination host/port filtering
  - URL, pattern/keywords, rewrite/truncate rules, protocol content filters
  - Detect ActiveX and Java applets; configure specific applets as trusted
    - Remove others from the HTML code
- Keyword detection
  - Prevent classified material from leaving the organization
  - Prevent banned content from leaving or entering an organization

Deep Content Inspection (DCI)

Deep Packet Inspection evolves to Deep Content Inspection

- Deep Packet Inspection systems
  - Rely on pattern matching and reputation lookup
  - Usually limited to buffering a small set of packets for a stream
- Deep Content Inspection systems
  - Unpacks encoded data
    - Example: base64-encoded MIME data in web and email content
    - Signature matching, compliance analysis (including data loss prevention)
    - Behavior analysis via correlation with previous sessions

The difference is largely marketing on levels of application-layer inspection that take place

Design Challenges With DPI

- DPI matches IP packet data against known bad patterns
- This must be done at network speeds
  - DPI hardware can only hold a limited number of packets for matching
  - DPI hardware can only store a limited amount of malware patterns

IDS/IPS
Intrusion Detection/Prevention Systems

IDS/IPS systems are part of Application-layer firewalls

Identify threats and attacks

- **IDS**: Intrusion Detection System
  - Monitor traffic at various points of the network and report problems
- **IPS**: Intrusion Prevention System
  - Sit in between two networks & control traffic between them (like a firewall)
  - Enforce admin-specified policy on detection of problems

Types of Systems

- **Protocol-based**
  - Signature-based
    - We know what is bad; anything else is good
  - Anomaly-based
    - We know what is good; anything else is bad

Signature-based IDS

Don't search for protocol violations but for possible data attacks

Match patterns of known "bad" behavior

- Viruses
- Malformed URLs
- Buffer overflow code

Need a database of known protocol attacks & malware

- Signature = data segments & order of packets that make up the attack
- Only detects known attacks

Anomaly-based IDS

Search for statistical deviations from normal behavior

Establish baseline behavior first

Examples:

- Port scanning
- Imbalance in protocol distribution
- Imbalance in service access

Challenge

- Distinguish anomalies from legitimate traffic

Application proxies

Proxy servers

- Intermediaries between clients and servers
- Stateful inspection and protocol validation

Application proxies

Dual-homed host

Bastion host

Protocol-Based IDS

Reject packets that do not follow a prescribed protocol

- Permit return traffic as a function of incoming traffic
- Define traffic of interest (filter), filter on traffic-specific protocol/patterns

Examples

- **DNS inspection**: prevent spoofing DNS replies; make sure they match IDs of sent DNS requests
- **SMTP inspection**: restrict SMTP command set (and command count, arguments, addresses)
- **FTP inspection**: restrict FTP command set (and file sizes and file names)
Firewall Challenges

Deperimeterization
Boundaries & access between internal & external systems are harder to identify

- Mobile systems
- Cloud-based computing
- USB flash memory
- Web-based applications

Deperimeterization
Boundaries & access between internal & external systems are harder to identify

- Mobile systems
- Cloud-based computing
- USB flash memory
- Web-based applications

Host-based (personal) firewalls

- Run on the user’s systems, not as dedicated firewalls
- Manage network-facing effects of malware
  - Allow only approved applications to send or receive data over the network
- Problem
  - If malware gets elevated privileges, it can reconfigure or disable the firewall
- Personal IDS
  - E.g., fail2ban on Linux
  - Scan log files to detect & ban suspicious IP addresses
  - High number of failed logins, probes, URLs that try to target exploits

Intrusion detection & prevention problems

- There’s a lot of stuff going on
  - People visit random websites with varying frequencies
  - Software accesses varying services
  - Buggy software may create bad packets
  - How do you detect what is hostile?
- Attack rates is miniscule … compared to legitimate traffic
  - Even a small % of false positives can be annoying and hide true threats
- Environments are dynamic
  - Content from CDNs or other large server farms has a broad range of IP addresses
  - Malicious actors can coexist with legitimate ones

Intrusion detection & prevention problems

- Encrypted traffic cannot be easily inspected
  - Just because you visit a web site using HTTPS doesn’t mean the site is secure ... or hasn’t been compromised
- Packet inspection is limiting
  - You may need to extract data from multiple packets
  - You may need to reconstruct sessions
  - Both of these are time consuming and can affect performance
- Threats & services change
  - Rules must be updated

Summary

<table>
<thead>
<tr>
<th>Firewall (screening router)</th>
<th>1st generation packet filter that filters packets between networks. Blocks/accepts traffic based on IP addresses, ports, protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stateful inspection firewall</td>
<td>2nd generation packet filter – like a screening router but also considers TCP connection state and information from previous connections (e.g., related ports for services)</td>
</tr>
<tr>
<td>Deep Packet Inspection firewall</td>
<td>3rd generation packet filter – examines application-layer protocols</td>
</tr>
<tr>
<td>Application proxy</td>
<td>Gateway between two networks for a specific application. Prevents direct connections to the application from outside the network. Responsible for validating the protocol.</td>
</tr>
<tr>
<td>IDS/IPS</td>
<td>Cuts usually do what a stateful inspection firewall does + examines application-layer data for protocol attacks or malicious content. Usually a part of Deep Packet Inspection firewalls</td>
</tr>
<tr>
<td>Host-based firewall</td>
<td>Typically screening router with per-application awareness. Sometimes includes anti-virus software for application-layer signature checking</td>
</tr>
<tr>
<td>Host-based IPS</td>
<td>Typically denies real-time blocking of remote hosts performing suspicious operations (port scanning, ssh logins)</td>
</tr>
</tbody>
</table>
DDoS: Distributed Denial of Service

- Compromise machines and create a botnet
  - Systems contact a command & control server for directions
  - Use amplification techniques to generate a lot of traffic for targets
- Exploit services that generate a lot of traffic to a small query
- DNS amplification:
  - Small UDP query with forged source address results in large response
- Some targets were too huge to hurt with traffic
  - Amazon, Google, sites using CDNs such as Akamai
- Vast quantities of compromised systems reduce need for amplification
  - Create a botnet of millions of systems

Dealing with DDoS

- Really difficult in general
- Bandwidth management routers
  - Either in data center or ISP
  - Limit outbound or inbound traffic on a per-IP basis
- Detect DNS attack and set null routing
  - Traffic to attacked DNS goes nowhere
- Egress filtering by ISPs
  - Attempt to find malicious hosts participating in DDoS or sending spam
- Identify incoming attackers & block traffic at firewall
  - Difficult with a truly distributed DDoS attack

The end