Network Security

{wnicole}@ethz.ch

ETH Zürich, HS 2020

This is a summary for the course *Network Security* at ETH Zurich.

We do not guarantee correctness or completeness, nor is this document endorsed by the lecturers. Feel free to point out any erratas.

Contents

1 Refreshers

1.1 Crypto Refresher

1.1.1 Definitions

- **Secrecy:** Keep data hidden from unintended receivers.
- **Confidentiality:** Keep someone else's data secret.
- **Privacy:** Keep data about a person secret.
- **Anonymity:** Keep identity of a protocol participant secret.
- **Data Integrity:** Ensure data is correct (no unauthorized / improper changes).
- **Entity Authentication/Identification:** Verify the identity of another protocol participant.
- **Data Authentication:** Ensure that data originates from claimed sender.

1.1.2 Symmetric Cryptography (Shared/Same-Key)

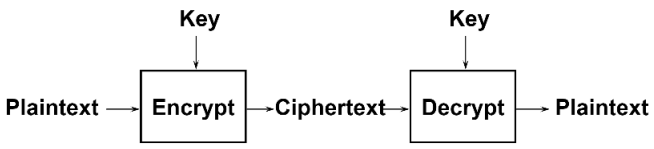

Figure 1: Symmetric encryption primitives.

For a plaintext enrypted with key K, we write $\{\text{plaintext}\}_K$. The main challenge of symmetric cryptography is the distribution of the secret key (a confidential and authentic channel is required).

Stream / State Ciphers Plaintext digits combined with a pseudorandom cipher digit stream (keystream) to encrypt them one at a time with the corresponding keystream digit (typically with an XOR operation).

Keystream Utilize a pseudorandom generator (PRG) to generate a keystream from a seed. E.g. Use shared key k and initialization vector IV for the seed, share cipher text and IV (ciphertext = plaintext $\oplus$ PRG(k, IV).

Initialization Vector / Nonce Pseudorandom fixed-size input to a cryptographic primitive. Best if non-repeating in repeated encryption applications.

One-Time Pad Stream Cipher Plaintext encryption using a one-time pre-shared key with the same size (or longer than) the message being sent. Impossible to break if: 1) key is truly random and 2) key is never (partially) reused.

ChaCha / Salsa Stream Cipher A 512-bit keystream is generated with a 256-bit key and a 64-bit nonce using add-rotate-XOR operations.

Keystream Reuse Attack If A and B are same-length plaintexts encrypted with the same key K, we have (with C being the key stream):

$$\{A\}_K = A \oplus C \text{ and } \{B\}_K = B \oplus C$$

$$\{A\}_K \oplus \{B\}_K = (A \oplus C) \oplus (B \oplus C) = A \oplus B \oplus C \oplus C = A \oplus B$$

Even without knowing A or B specifically, both plaintexts can be easily derived / analyzed.

Ciphertext Modification Attack Changing digits of the ciphertext will alter the corresponding values in the plaintext after decryption. Authenticity of ciphertext is required to defend against this attack.

Block Ciphers Blocks (fixed-length groups of bits, last block padding if needed) are encrypted using a symmetric key each (one-to-one mapping). E.g. Advanced Encryption Standard (AES).

Mode of Operation Describes how to repeatedly apply a cipher's single-block operation to encrypt the data. There are two categories: confidentiality-only (ECB, CBC, CTR, etc.) and authenticated encryption (AE) that combines confidentiality and authenticity. Integrity protection is an entirely separate goal seen in AEAD only.

Electronic Code Book (ECB) Natural approach - split plaintext into blocks and use the same key for each block. Has the disadvantage that equal blocks correspond to equal ciphertexts - lack of diffusion. ECB does not hide data patterns well (esp. in images with large areas of uniform colors).

Cipher Block Chaining (CBC) Apply XOR to each block with previous ciphertext before encrypting. Use an IV for the first block. Each ciphertext block depends on all plaintext blocks processed up to that point. Cannot be parallelized and message must be padded to a multiple of cipher block size. Plaintext block can be recovered from two adjacent blocks of ciphertext (allows for parallelized decryption).

Counter Mode (CTR) Turn block cipher into a stream cipher. Generates the next keystream block by encrypting successive values of a counter (any function that does not repeat for a long time, e.g. increment by one counter). Use an IV for the first block and increment for next blocks. Has the same vulnerabilities as any other stream cipher.

Authenticated Encryption Ensures confidentiality and authenticity of data. E.g. Encrypt-then-MAC (EtM), Encrypt-and-MAC (E&M), MAC-then-Encrypt (MtE) - see Figures 2, 3 and 4.

Message Authentication Code (MAC) Cryptographic checksum for message authentication and integrity. Can only be calculated with a shared secret key ($\text{MAC}_K(\text{M})$).

Authenticated Encryption with Associated Data (AEAD) Allows a recipient to check the integrity of both the encrypted and unecrypted message (on top of confidentiality and authorization). Based on Encrypt-and-MAC (E&M).

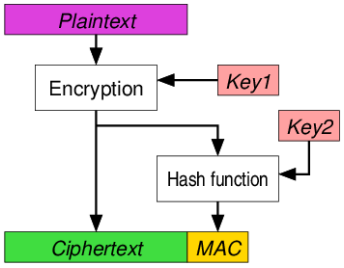

Figure 2: EtM approach.

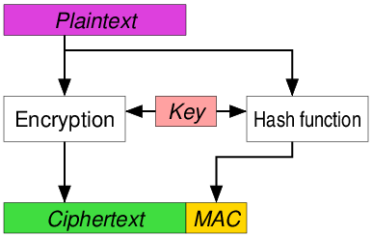

Figure 3: E&M approach.

Galois Counter Mode (GCM) AEAD based on a block cipher in CTR mode (block size of 128 bits) and Galois MAC - mostly used with AES (AES-GCM). See Figure 5.

1.1.3 Asymmetric / Public-Key Cryptography / PKI

Cryptographic primitives using pairs of keys (public and private). The pairs are generated using cryptographic one-way functions. Can be used to establish a secret symmetric key.

Messages are encrypted using the receiver's public key and can only be decrypted with the receiver's private key. To ensure authenticity, a message can additionally be signed with the sender's private key and verified with the sender's public key by the receiver.

The main challenge is the authentication (i.e. verify identities) of the key (an authentic channel is required).

Diffie-Hellman Key Exchange A method to securely exchange a key over a public channel based on the discrete logarithm problem (see Figure 6) - the key (or a derived key) can then be used to encrypt subsequent communication using a symmetric key cipher. DHKE Is susceptible to man-in-the-middle attacks in which an attacker impersonates either receiver to each sender, resulting in two different established keys.

Discrete Logarithm Problem Given x, g and p, find a with $g^a \bmod p = x$.

RSA Algorithm Same as with Diffie-Hellman, just in a different way (see Figure 7).

Encrypted Key Exchange (EKE) Both ends share a password P and want to authenticate each other along with establishing a shared secret key (see Figure 8).

At least one party encrypts an ephemeral (one-time) public key using a password and sends it to the other party. Other party decrypts public key and uses it to negotiate a shared key with the former.

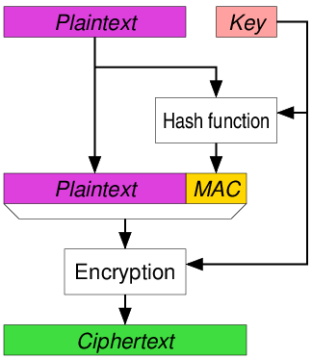

Figure 4: MtE approach.

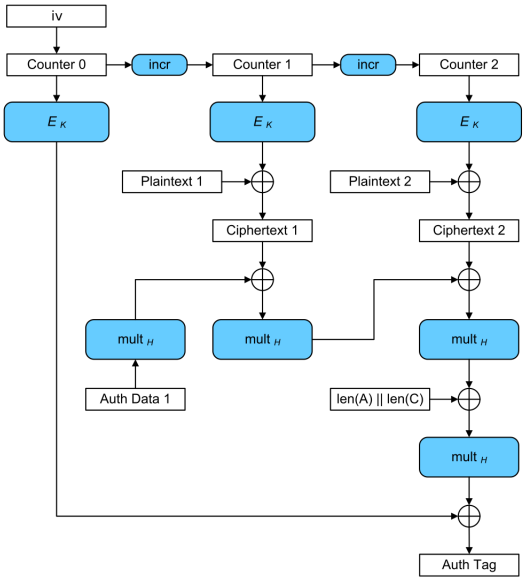

Figure 5: GCM with only two plaintexts.

Authentication vs. Signature Authentication enables a receiver to verify the origin of a message, but it cannot convince a third party of the origin. A signature allows for the verification of origin and will convince a third party of it as well (a signature provides authentication).

1.1.4 Hash Functions

Cryptographic Hash Functions Map an arbitrary-length input into a finite length output with the following properties:

- **One-way:** given $y = H(x)$, one cannot find x' s.t. $H(x') = y$ - i.e. result cannot be reproduced without knowing original input.
- **Weak collision resistance:** given x, one cannot find $x' \neq x$ s.t. $H(x) = H(x')$.
- **Strong collision resistance:** one cannot find $x' \neq x$ s.t. $H(x) = H(x')$, i.e. different inputs will not produce the same output.

They can be used for digital signatures, MACs and other forms of authentication. Also used for fingerprinting (unique IDs, no duplicates) and as checksums (check integrity). Most common cryptographic hash functions include MD5, SHA-1, SHA-2 and SHA-3.

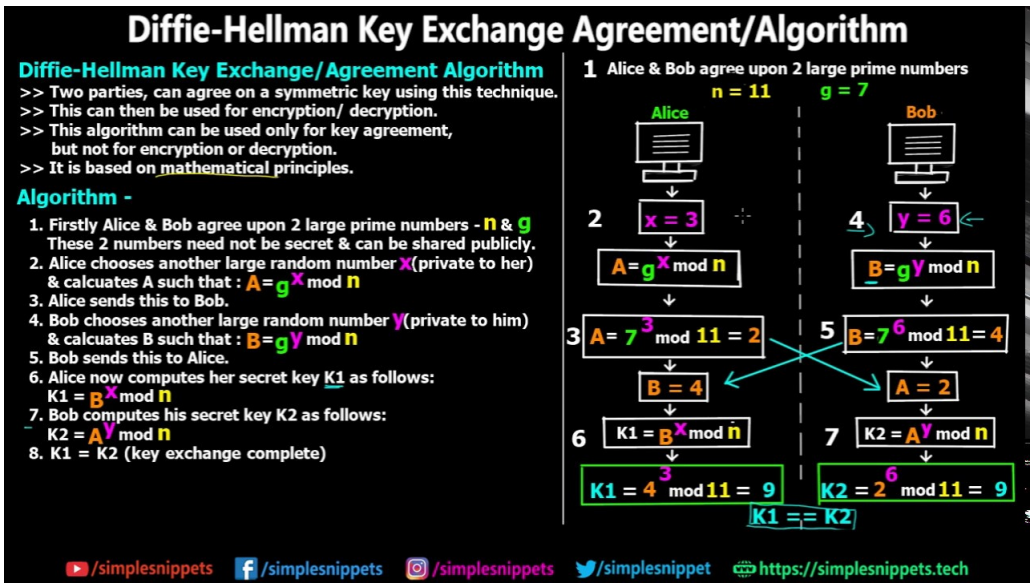

Figure 6: Diffie-Hellman key exchange.

SHA-1 is known for collision attacks. E.g. Git only uses SHA-1 value to identify commits - attacker can commit good code and distribute a repo version with bad code that has the same SHA-1 value.

One-Way Hash Chain A method to produce many one-time keys from a single key. Pick a random starting value r_n and a public one-way hash function, continuously apply function to resulting values where r_0 is the last result. Reveal results in ascending order $(r_0, r_1, r_2, ..., r_n)$. Receiver can authenticate r_i with r_j where $i < j$ and $r_i = F(r_j)$, i.e. when receiving a value, receiver can authenticate previous value.

Merkle Hash Trees (Binary Hash Chain) Efficient and secure verification of large amounts of data. Data is divided into blocks with each block labelled with its cryptographic hash value (leaf nodes). Every non-leaf node is the hash value of the concatenated hash values of its two children (if binary).

For two Merkle Trees on the supposedly same data, the value of the root node has to be the same. Else, traverse tree downwards to find conflicting data blocks.

To demonstrate that a leaf node is part of a binary tree, it is required to compute a number of hashes proportional to the log of leaf nodes - this allows for partial checking (as compared to hash lists where we would need the full file). Usually, the root hash received from a trusted source is compared to a hash tree from a potentially untrusted source.

It is impossible that the same two child values produce different parent values. In practice, it is possible that different children can produce the same parent value (collision).

1.2 Network Refresher

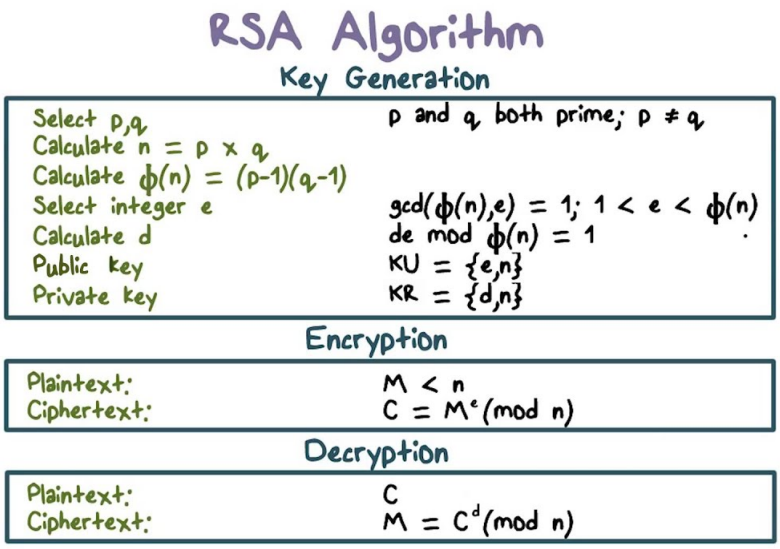

Figure 7: RSA key generation and de/encryption.

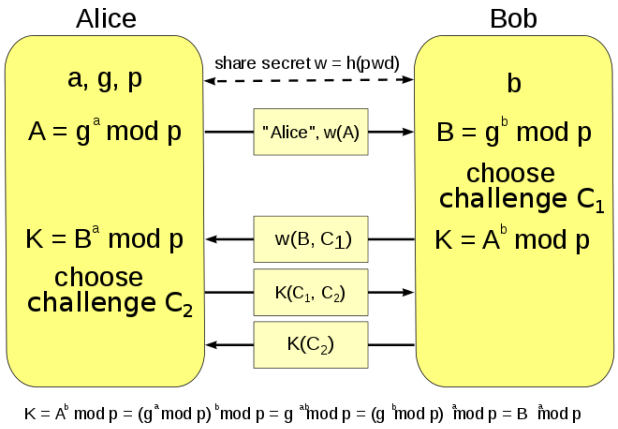

Figure 8: DH-EKE scheme.

2 SSL/TLS Public Key Infrastructure

Goal of TLS Achieve secure internet communication by preventing eavesdroppers from learning sensitive information and enabling entity and message authentication.

TLS Challenges

- No additional latency of contacting servers during TLS handshake caused by authentication.
- Keys should be immediately usable and verifiable after initial registration.
- Users should not be concerned with checking the legitimacy of a certificate.
- Covering the entire certificate life cycle.

Certificate Authority (CA) Entity (trusted third-party) that issues digital certificates, which certify the ownership of a public key by the uniquely named subject.

(Public-Key) Certificate The format of such certificates is specified by the *X.509* or *EMV* (proprietary - used for ATMs and the like) standard. Certificates can be revoked and renewed (challenge of a distributed system: consistency, availability and partition tolerance - CAP theorem).

A certificate can be signed by a CA or self-signed - (changes the trust model, considered unsafe for public-facing services, suitable for internal sites or testing environments).

Trust Anchor / Root Self-signed certificates of public keys that are allowed to sign other certificates (trust assumed, not derived). One anchor can involve multiple entities. Involved in the whole chain of trust and used during certificate path validation (certificate hierarchy). With cryptographic operations, trust can be transferred from one entity to another.

Root certificates are only used to sign intermediate certificates since they need to be as safe as possible. Revoking even one of them can make many certificates invalid if they sign more. Roots are kept offline and are used as rarely as possible.

PKI Alternatives If one of the many trusted CAs is compromised, security of all digital certificates is susceptible to attacks. One alternative architecture is Attack Resilient PKI (ARPKI). To impersonate a domain, a large number of entities need to be compromised. Another alternative is a web of trust = distributed PKI as seen in Pretty Good Privacy (PGP).

ARPKI

Web of Trust / PGP Decentralized PKI where users establish direct trust with each other (private and public key system). Indirect trust to a destination can be established by trusting someone that directly trusts destination (think if a network graph). There are no CAs and therefore no single point of failure stemming from a compromised CA hierarchy.

PGP uses this model to provide cryptographic privacy and authentication for data communication. It is used for signatures and encryption.

Trust Agility The ability for the user to choose which entities to trust. User can also specify to require additional certification for entities not covered by chosen group of trust anchors.

Levels of Trust Least to most trusted:

- **No SSL/TLS:** domain/website served via HTTP.
- **Domain Validation (DV):** Domain name is validated, applicant only has to prove control over it in some way (e.g. publish a DNS TXT record). Site is secure, not necessarily the entire business, controlling applicant can be malicious.
- **Organisation Validation (OV):** CA carries out some vetting of the organisation (verify existence of company and domain name, verification phone call, etc.).
- **Extended Validation (EV):** CA undertakes comprehensive vetting process of the organisation (verify legal and physical existence of organisation, identity matches official records, exclusive rights to the domain, etc.). Only a subset of CAs can issue EV certificates.

HTTP Strict Transport Security (HSTS) Allows web servers to declare that browsers / user agents should automatically interact with it using only HTTPS connections. HSTS helps to protect websites against MITM-attacks (i.e. protocol downgrade, cookie hijacking a.k.a stealing session keys in HTTP, etc.).

HTTP Public-Key Pinning (HPKP) (deprecated) Allows HTTPS websites to resist impersonation by attackers using misissued/fraudulent certificates. A set of hashes of public keys valid for a given time is delivered to the client (web browser) which must appear in the certificate chain of future connections to the same domain name. During their validity time, a client expects to see one or more of those public keys in its certificate chain.

Since HPKP is trust on first use (TOFU), meaning the provided hashes are trusted, it can still be susceptible to a MITM attack if the first policy received comes from an attacker. Furthermore, weak hashes allow for easy collisions.

Careful: when renewing the certificates for a domain, new hashes (with a new age) have to be distributed to not DoS oneself.

Online Certificate Status Protocol (OCSP) Protocol to obtain the revocation status of an X.509 certificate - an alternative to certificate revocation lists (CRL). OCSP responders (servers) respond to requests to verify the status of a certificate to check if it is still valid. Vulnerable to replay attack where an attacker captures the "still good" response and replays it at a later time where the certificate might already be revoked.

OCSP Stapling Standard for checking the revocation status of X.509 digital certificates. The presenter of a certificate can bear the resource cost involved in providing OCSP responses by appending (= stapling) a time-stamped OCSP response signed by the CA to the initial TLS handshake - eliminating the need for the client to contact the CA.

Original OCSP introduced a significant cost for CAs because they have to respond to numerous OCSP requests. Furthermore, privacy issues were introduced with the need to contact a third-party and not receiving a response can be annoying.

DNS-Based Authentication of Named Entities (DANE) Protocol to bind X.509 certificates to domain names using Domain Name System Security Extensions (DNSSEC). Allows to authenticate TLS entities without a CA.

Certificate Transparency (CT) A (proposed) standard for monitoring and auditing digital certificates. It creates a system of public logs recording all certificates issued by publicly trusted CAs. This allows for efficient identification of mistakenly / maliciously issued certificates (which is usually very slow). Makes it impossible for a certificate to be issued for a domain without the domain owner knowing.

In a CT log, new certificates are appended to an ever-growing signed (by log server) Merkle Hash Tree. Each appended certificate must have a valid signature chain. Monitors (CAs) check the log servers for misbehavior and request the addition of a newly issued cert. Auditors (clients) verify certificate existence by checking the log servers and exchange info with the monitors on the log server status (to ensure that it is not compromised).

Certificates stay on the log forever (even revoked ones). Merkle Tree insertions are easy but immediate deletion is not. System might implement revocation transparency.

Often used together with OCSP stapling in which a certificate presenter has to add certificate to a public log and the CT log signature needs to be stapled to signature.

3 More on TLS

3.1 TLS Overview

High Level Goals TLS aims for security in face of an attacker with complete control of the network. The only requirement for the underlying transport is a reliable and in-order data stream (e.g. TCP) - TLS is between the transport and application layer.

- **Entity Authentication:** Server side is always authenticated, client side optional. Authentication via asymmetric crypto (signatures) or symmetric pre-shared key (unique for each session).
- **Confidentiality:** Data only visible to the endpoints, but TLS does not hide length of data (padding possible).
- **Integrity / Reliability:** Data cannot be modified without detection. Guarantees also cover reordering, insertion and deletion.

Part 1: Handshake Protocol Used by client and server to negotiate cipher suite[1] (set of algorithms to secure the connection, e.g. key exchange, encryption, MAC, etc.), authenticate each other (certificates and public keys) and establish session keys used in the Record Protocol (see below).

Establishing Session Keys The client either encrypts a random number with the server's public key, with which both parties generate a unique key or the client uses a Diffie-Hellman key exchange (see previous Section). The second option has the property of forward secrecy if ephemeral public values are used.

Forward Secrecy Future disclosures of encryption keys cannot be used to decrypt communications recorded in the past. More specifically, session keys cannot be compromised even if long-term secrets used in the session key exchange are compromised. Additionally, compromising unique session keys should not affect any other communications encrypted with a different session key. Assumes passive attacks, i.e. does not protect against an active MITM attack.

An encryption system has the property of forward secrecy if plain-text (decrypted) inspection of the data exchange that occurs during key agreement phase of session initiation does not reveal the key that was used to encrypt the remainder of the session.

E.g.: In HTTPS, the long-term secret is typically the private signing key of the server.

Part 2: Record Protocol Using the session keys established in part 1, the application layer communication is guaranteed to be confidential and reliable.

The Record Protocol provides a stream-oriented API (fragmentation/coalescence of data allowed). A record is a TLS data unit, which is a fragment of the data stream.

[1] Click link for a list of all 337 cipher suites and their security status.

Cryptographic Protections in the Record Protocol

- Data origin authentication.
- Data integrity using a MAC.
- Data confidentiality using a symmetric encryption algorithm (integrity and encryption combined in TLS 1.3 by use of AEAD).
- Prevention of reflection attacks by key separation (different symmetric keys in different directions).

Reflection Attack Attacking a challenge-response authentication system that uses the same protocol in both directions by tricking the target into providing the answer to its own challenge.

TLS 1.0 (1999) First version of TLS (after migrating from SSL) and deprecated in 2020 due to security flaws.

TLS 1.1 (2006) Fixed the security flaws of version 1.0, such as protection against cipher-block chaining attacks.

TLS 1.2 (2008) Changed hash algorithms used for the protocol, enhanced cipher suite negotiation abilities between client and server, support for authenticated encryption ciphers (see AE(AD)), addition of AES cipher suites, etc. In 2011, backward compatibility with SSL was removed to prevent version fallback. See Figure 9 for the architecture of the protocol.

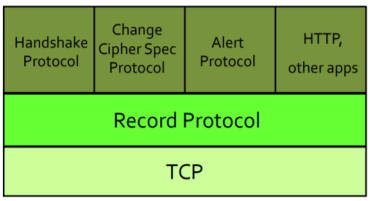

Figure 9: TLS 1.2 protocol architecture.

TLS 1.2 Handshake When using TCP, the full handshake is a 3-RTT protocol (see Figure 10).

TLS 1.3 (2018)

- Not yet widely used.
- Significant change to the way of coordinating the cipher suites between machines (part of the Handshake Protocol) by reducing the number of messages needed (cannot be used for version 1.2 and vice versa - the numbers of suites is reduced from 337 to 5).
- The key agreement and authentication algorithms are separated from the cipher suites.
- Mandates forward secrecy.
- Dropped support for insecure obsolete features such as compression (enabled *CRIME* attack), renegotiation, non-AEAD ciphers (previous versions support MtE), etc. and introduced further security measures.

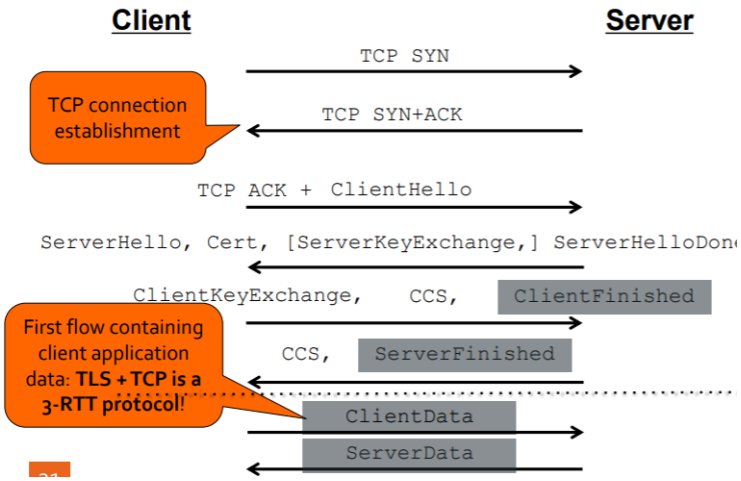

Figure 10: The full TCP + TLS 1.2 handshake.

- Encrypts all handshake messages after the *ServerHello*. Compared to previous versions where the complete handshake is in the clear, version 1.3 encrypts almost all handshake messages with a separate key, providing security to both passive and active attacks on both ends.

TLS 1.3 basically wants to mimic QUIC by achieving the same RTT profile (excluding TCP overhead since it is unavoidable for TLS). QUIC offers a native 1-RTT handshake and a 0-RTT mode.

See Figure 11 for the architecture of the protocol.

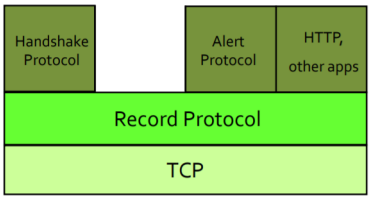

Figure 11: TLS 1.3 protocol architecture.

Datagram TLS (TLS over UDP) Prevents eavesdropping, tampering or message forgery of UDP connection. Has to deal with fragmentation and packet reordering / loss. Avoids the *TCP meltdown problem* when being used to create a VPN tunnel[2].

3.2 TLS 1.3 In More Detail

TLS 1.3 Record Protocol

- Each fragment is encoded into payload, ctype (single byte representing content type, i.e. handshake message, alert message or application data) and optional padding (used to hide true length of fragments).
- Encoded fragments are then encrypted using AEAD and appended to a record header.
- For the AEAD encryption, a nonce is created by XORing the SQN and an IV. The 64-bit sequence number (SQN) is incremented for each record sent on a connection and maintained at each end of the connection (not included in header). The IV is a fixed per-TLS connection pseudorandom

[2] *Why TCP over TCP Is A Bad Idea*

value derived from secrets established during the Handshake Protocol. XOR (masking) ensures that the nonce sequence is unique for each connection. This facilitates analysing security in a multi-connection setting.

- The record header contains a 1-byte dummy type field, a 2-byte legacy version field (negotiated during handshake) and the length of the AEAD ciphertext (2 bytes).
- AEAD-decryption failures are fatal and will lead to the connection being torn down and the key material being thrown away.

Attacks not Prevented by TLS 1.3 Record Protocol

- Truncation attacks on the stream of records.
- Application layer confusion.
- Timing attacks on the padding scheme.

TLS 1.3 Handshake Protocol

- Full handshake in 1 RTT (compared to 2 RTTs in TLS 1.2 and earlier resp. 3 RTTs if TCP is counted).
- Achieved by only ever doing DH (elliptic or finite-field) and client speculatively sending several DH shares in supported groups (only a few) - server then picks one, replies with its share and Record Protocol keys can be derived.
- When resuming a previously established connection, a 0-RTT handshake is possible since shared state is kept with which a PSK can be derived.
- TLS 1.2 and earlier had the complete handshake in the clear, 1.3 encrypts almost all handshake messages and derives separate key to protect them (security for passive and active attacks).

3.3 Future of TLS

4 IPv6 Security

Guest lecture.

4.1 Introduction

Address Space IPv6 uses 128-bit addresses, which theoretically allows for 2^{128} unique addresses (compared to 64-bit in IPv4). An address is represented as eight colon-separated groups of four hexadecimal digits (can be shortened by excluding the zeros). E.g. *2001:db8::8a2e:370:7334*.

Link-Local Address (LLA) An address that is only valid for communications within the network segment/broadcast domain that a host is connected to. In IPv6, they are assigned from the block *fe80::/10* (prefix, with a 64-bit suffix computed by host itself) and every host is required to have at least one / one for each network interface. An LLA is unique in the LAN (also named unique local address, ULA). LLAs can be translated to global addresses using a NAT.

To not leak any information, LLAs do not embed the host's MAC address anymore and are instead generated via algorithms using some kind of randomness. Knowing a MAC address allows for physical attacks.

Link-Layer Address (MAC) To map IP to MAC addresses, IPv6 uses the Neighbor Discovery Protocol (NDP) instead of ARP since IPv6 does not support the broadcast addressing method (as used by ARP). Nodes using the requested IP address respond to neighbor solicitation messages originating from an asking host with their LLAs.

This has the potential for a DoS attack by overflowing a host's buffer or overwriting real entries.

Router Solicitation By sending a multicast ICMPv6 router solicitation message to the all-routers group, a host learns about the network from neighbouring routers (router advertisement) and can establish a globally unique address with an appropriate unicast network prefix. Router advertisements also include further information, i.e. if and how DHCP should be used by the host.

Multicast Groups IPv6 does not implement IP broadcast with a special broadcast address. Using multicast groups, a message can be broadcasted to all link-local nodes (*ff02::1*) or all link-local routers (*ff02::2*).

4.2 IPv6 Security

Duplicate Address Detection (DAD) DoS Hosts use DAD to assign themselves unique LLAs. To ensure they are unique, hosts ask all nodes of a LAN if an LLA is already in use. This allows for an easy DoS by simply replying that the address is already in use to each DAD request.

Shadow Networks Data flowing through new IPv6-enabled connections and onto the existing IPv4 network while the IPv4 security in place is unable to identify this new type of traffic (e.g. traffic is not subject to firewall rules, etc.). This is a problem if a network admin is not aware of new IPv6 ability.

In general, IPv6 connectivity cannot be prevented as long as a node is part of a bi-directional communication. Such a channel can be used as a tunnel.

Tunneling The act of encapsulating IPv6 packets in IPv4 packets to allow for IPv6 connectivity anywhere. Not used much. E.g.: *Teredo*, *6to4*, etc.

Injecting IPv6 Addresses In a LAN without an IPv6 router, a host could be made to act like one. All IPv6 capable hosts automatically assign themselves an IPv6 address. With this, the host can easily enact a MITM attack and bypass any IPv4 firewalls (via tunneling). If a LAN already hosts IPv6 routers, one can simply setup a router with a higher priority since router advertisements include a priority field.

Such rogue router advertisements can be prevented via filtering (RA and DHCPv6 messages).

Filtering

Fragmentation Attack IPv4 fragments packets, requiring routers to store and/or re-assemble packets. By sending bogus or overlapping fragments, a router can crash (memory exhaustion). IPv6 does not support fragmentation in the network - the work is shifted to the end hosts (routers return ICMP6 messaged indicating that a packet is too big).

Internet Protocol Security (IPsec) Authentication and encryption of IP packets (mainly used for VPNs). Part of the IPv6 specification but not implemented in practice (too complex (NATs), other methods of security) - only a recommendation.

NAT Is Not Security (NINS) NAT maps IP addresses 1:1 / 1:n or to protocol ports (PNAT). It is not to be used as a (stateful) firewall, since access from outside is possible if the table entries are known.

Conclusions

- IPv6 is not more or less secure than IPv4.
- Attacks are similar to the ones on IPv4.
- Networks need to be prepared to handle IPv6 to avoid security holes.

5 Virtual Private Networks

5.1 Overview

VPN A VPN creates a secure channel between two private networks over an untrusted, public network (e.g. the Internet) - this provides security on the link / network layer. In addition, functionality and management of the private network can be transferred to the established VPN.

A VPN can connect physically separated networks (site-to-site), connect a remote host to a company/university network (host-to-site), circumvent censorship, avoid tracking, hide a user's IP address, spoof locations, allow access to restricted content, etc. A VPN provider has access to metadata of all traffic - anonymity is not guaranteed!

Part 1: Set-Up Phase The tunnel endpoints authenticate each other and set up keys (similar to the TLS handshake).

Part 2: Tunneling Phase Packets are encapsulated and decapsulated at the endpoints. In addition, the original data is encrypted and authenticated with a MAC.

VPN Tunnel Properties A VPN tunnel holds similar security properties as the TLS Record Protocol (source authentication, data integrity, confidentiality, replay suppression, etc.). However, some tunneling protocols do not provide encryption/authentication.

VPN vs. TLS

- VPN protects all kinds of traffic (e.g. DNS requests, etc.).
- Services in private networks / behind firewalls can be accessed with VPNs.
- VPN data is only secure in the tunnel. TLS protects data end-to-end.
- VPN servers can see unencrypted traffic (here, TLS is still necessary).
- A web server cannot be authenticated with VPN, only the tunnel endpoint.
- TLS does not authenticate clients (by default).
- VPN utilize many different protocols.
- TLS is commonly used on top of a VPN.

Performance / Availability VPNs introduce overhead with additional cryptographic operations, potential detours and limited bandwidth at VPN servers. There is no built-in defense against (D)DoS and routing attacks, therefore VPNs do not provide a higher availability. Packet filtering targeting VPN packets would need to target all VPN packets since content is hidden.

VPN vs. VLAN

- A VPN connects two different networks, resulting in one virtual network over multiple physical ones.
- VLANs set up multiple isolated virtual networks (logical groupings) on a single physical infrastructure.
- VLANs are often used in cloud-computing environments to isolate communication between VMs or to form multiple broadcast domains in a network with frequent broadcasts.
- VLANs do not provide authentication.
- VLANs are L2 constructs, analogous to IP subnets which are L3 constructs. There is often a one-to-one relationship between a VLAN and an IP subnet (however, it is also possible to have multiple subnets on one VLAN).
- A virtual extensible LAN (VXLAN) combines both features (VLAN and VPN).

Implementing VPN Functionality There are many different protocols and applications that implement VPN functionality. They can differ in authentication mechanisms (PSK, PKI, etc.), the tunnel protocol (custom ones, SSTP, etc.), the inner protocol = outermost header of an encapsulated packet (network vs. link layer VPN - e.g. IPsec L3, WireGuard L2, etc.) and of course how they're implemented (in user space, a kernel module, directly in hardware, etc.).

Endpoints at Hosts A VPN creates virtual network adapters (e.g. "eth2", "tun0", etc.) at the end hosts, one route with the public IP of the VPN server as a destination and a new default route to the private address of the VPN tunnel. They can be used like any other network adapter and one can either send all traffic through it or selectively (*split tunnel*) by choosing only a subset of IP prefixes that should use the VPN.

5.2 VPN Implementations

IPsec IPsec is a secure network protocol suite (very large, many options / modes) that authenticates and encrypts packets over an IP network - also supports data integrity and replay protection. There is mutual authentication between agents at the beginning of a session along with negotiation of keys to use during it. Problems include difficult configuration due to many options where some don't even provide security (no encryption and no authentication) and the possibility of insecure ciphers (similar to TLS problems).

Comparison to TLS: In HTTPS, client certificate authentication is rare while in IPsec it is the norm. This is because websites are usually publicly accessible while private networks apply strong restriction on authorized users. IPsec explicitly includes sequence numbers in its packets since IP is only best-effort (loss, reordering, duplication) while TLS maintains only local counters since it runs on top of TCP.

Typical Session: first set up a security association (SA) via Internet Key Exchange (IKE) and then encapsulate packets and tunnel them between SA endpoints with Encapsulation Security Payload (ESP).

IKEv2: first establish a temporary and anonymous key for the rest of the IKE exchange and then authenticate each other by sending identity, (certificate), sig (asym.) / MAC (sym.), etc. (see Figure 13). Passive attacks cannot uncover the identities but an active MITM attack might. In case of ephemeral DH values at the beginning, we have perfect forward security.

	TLS	IPsec
Key exchange	TLS handshake	Additional protocol: Internet Key Exchange (IKE)
Authentication	Typically only server Typically using RSA certificate	Server and client Many different authentication mechanisms
Underlying transport	Reliable (runs on top of TCP)	Best-effort (runs on top of IP)

Figure 12: Comparison of TLS and IPsec.

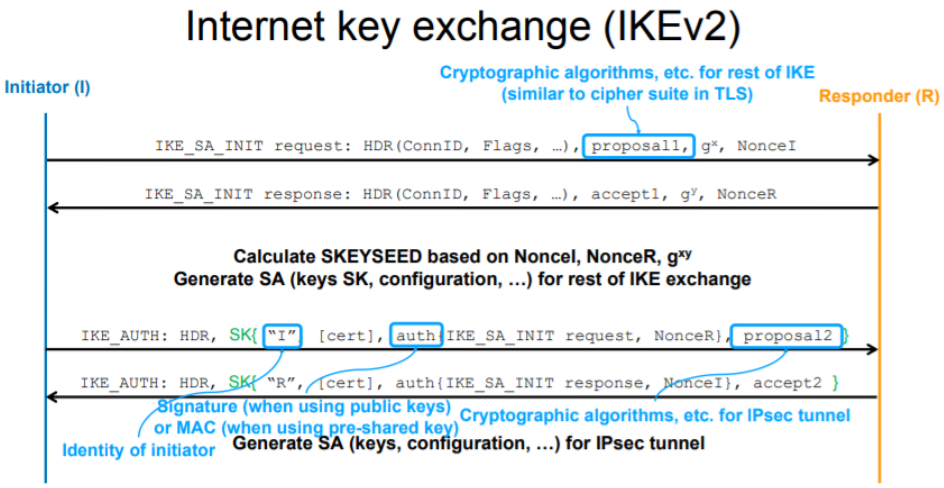

Figure 13: Simplified overview of the IKEv2.

ESP: the original IP packet is first padded with the ESP trailer that also includes the type of the original packet (appended at the end). It is then fully encrypted. Then, the ESP header is added which includes the SA identification and a sequence number (replay protection, nonce for certain encryption algos, etc.). Lastly, an Integrity Check Value (ICV) is created and appended at the end - it is a MAC over the entire packet. Finally, a new IP header is added.

Other Options: there can be additional messages in IKEv2 (EAP, cookies, etc.) and other modes like the transport mode for end-to-end connections or an Authenticated Header (AH) protocol instead of ESP if we only want authentication but no encryption.

WireGuard Much faster and smaller than other options (IPsec, OpenVPN, etc.) by removing cryptographic agility, i.e. only support chosen state-of-the-art primitives (single cipher suite) s.t. negotiation is simplified and insecure primitives are removed. Configuration is kept simple. Attack surface is minimal by keeping the codebase small (also makes formal verification easier). L3 packets are encapsulted and transformed into L4 UDP only packets.

Key Exchange and Authentication: the WireGuard handshake follows the Noise Protocol Framework built exclusively on elliptic-curve DH exchanges. First, each peer has a static key pair (public and private) and specify in configuration which public keys are authorized. Each peer then creares an ephemeral key pair (public and private). The symmetric keys are then derived from four DH combinations (two static public key and two ephemeral public keys). See Figure 14. Since WireGuard is connectionless, a series of timers (based on message counts and time) are used to steer handshakes, key renegotiation and session termination. Perfect forward security is guaranteed with strict key rotation timers and the ephemeral ECDH session key exchange.

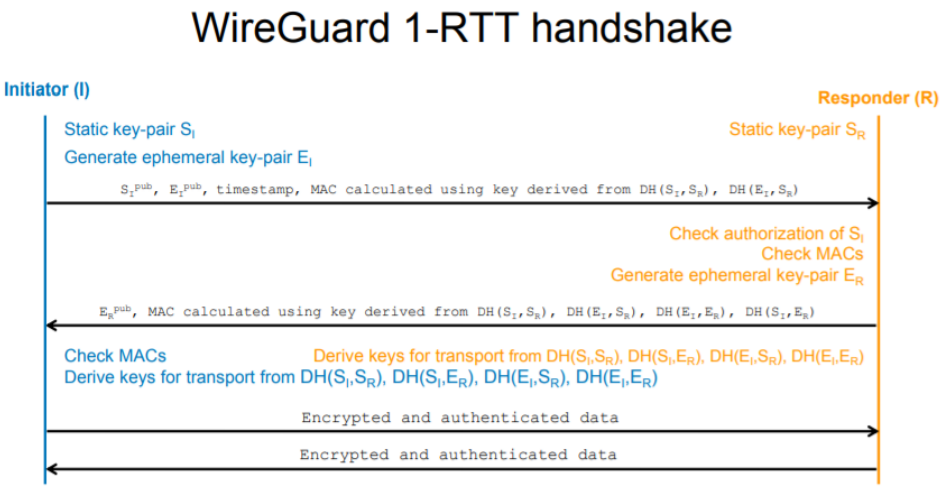

Figure 14: Simplified overview of the WireGuard handshake.

DoS Protection: When under load, WireGuard can choose to respond with a cookie instead of processing the handshake (expensive cryptography). With this cookie, the initiator uses it as a key for computing HMACs of their message (similar to IKEv2 cookie-mode). Difference to IKEv2 cookie-mode is that an additional MAC is required on the handshake message (responder public key as HMAC key). Responder can stay completely silent unless initiator knows its public key.

Point-To-Point Tunneling Protocol (PPTP) Comparison PPTP, L2TP, IPsec, etc.

Generic Routing Encapsulation (GRE)

Layer-2 Tunneling Protocol (L2TP)

OpenVPN

Secure Socket Layer (SSL)

Secure Socket Tunneling Protocol (SSTP)

5.3 Summary

- VPNs create secure channels on the network/link layer.
- VPNs and end-to-end security (TLS) complement each other.
- There are many different VPN protocols and applications.

6 Wireless LAN

Guest lecture.

7 Anonymous Communication Systems

7.1 Overview

Sender Anonymity Adversary knows / is receiver and may learn messages. The sender is unknown but provides a return address / token.

Receiver Anonymity Adversary knows / is sender and may choose the messages. The receiver is unknown and receives data via a hidden service (pseudonym is known).

Sender / Receiver Anonymity Set Set of all individual senders / receivers indistinguishable from the real sender / receiver. The smaller the set, the less anonymity.

Unlinkability Adversary knows senders and receivers but not the link in between. Multiple users need to be communicating for this to work. Anonymity results in unlinkability.

Unobservability Adversary cannot tell whether any communication is taking place. For wireless, use methods like DSSS. For wired, constantly send traffic. Unobservability results in anonymity.

Plausible Deniability Adversary cannot prove that any particular individual was responsible for a message / any action. Anonymity results in plausible deniability.

Thread Model(s) An adversary may have:

- Various degrees of control (local / global).
- Various types of (combinations of) control (network, compromised infrastructure, etc.). However, the infrastructure is never fully compromised.
- Passive or active behavior.

This results in unclear guarantees since the thread model is often not clearly specified.

7.2 Mechanisms for Anonymous Communication

- **Broadcast:** for wireless communication, guaranteed receiver anonymity. However, a sender can be de-anonymized via triangulation. Use DSSS if the destination is trusted.
- **Hijacked Connections:** using a burner phone or hacking a WiFi connection to impersonate an ID.
- **Proxy / VPN:** to hide content from a proxy, use layered encryption. Problems: proxy can see metadata and may be a single point of failure. Use a cascade (onion) of multiple proxies (each proxy only sees addresses of two neighbors, works as long there is one honest proxy, message / forwarding info is encrypted multiple times).

Mix-Nets To obscure in- and outgoing messages, each proxy performs batching (collecting several messages and forwarding them if specific threshold is met). Additionally, proxies should change the order of the messages (mix). All messaged need to be padded to a fixed length to make them indistinguishable.

Vulnerable to intersection attacks. Since users often only communicate with a small subset of other users, an adversary can register the sets of destinations each time a message is seen,i.e. long term correlation attack (also possible with inter-packet intervals). More effective attack: statistical disclosure.

Return addresses are prepared by senders to receive replies. Senders know what keys will be established with each threshold mix proxy.

Problem: high latency, bad for web browsing - see Onion routing below. Forward security is not guaranteed.

Cover / Dummy Traffic Achieves full unobservability and prevents statistical disclosure attacks both for sending and receiving. A receiver regularly tries to retrieve messages from a threshold mix proxy and downloads it if there is any, else dummy message is returned. Not possible on Onion Routing.

Onion Routing / Circuit-Based Anonymity Networks No batching and mixing and no cover traffic, only layered encryption. Onion routing is flow-based, a virtual circuit / tunnel (keys) is established once per flow using only symmetric key crypto and each packet is forwarded by relays (proxy nodes). This has a lower anonymity guarantee than the previously introduced mechanisms. Also, the threat model is constrained - we can only deal with local adversaries that cannot launch confirmation attacks.

A sender remains anonymous because each intermediary knows only the location of the immediately preceding and following nodes. Only the final node knows its location in the chain, others don't know if they're the first or immediate ones.

Circuit setup:

Direct circuit setup:

Telescopic circuit setup:

Data forwarding:

Circuit teardown:

For all points above, see extra notes.

Mix-Nets vs. Onion Routing

- **Forwarding System:** message-based vs. circuit-based.
- **Layered Encryption:** asymmetric vs. symmetric.
- **Forward Security:** no vs. yes in case of the telescopic setup.
- **Latency:** high vs. low / medium.
- **Guarantees:** strong vs. attacks possible for strong adversaries.

7.3 Attacks on Circuit-Based ACS

7.3.1 Traffic-Analysis Attacks

Passive Traffic Analysis Adversary observes edges of the network and records traffic patterns (flow length, bandwidth pattern, inter-packet timings). If measurements at two ends are similar / the same, an attacker could conclude that they are communicating. Real-time detection is challenging, more common to store and compare later (large storage needed).

Active Traffic Analysis Adversary actively modifies inter-packet timings (delaying / reordering packets). Packet drops possible but often detectable. If the clearly visible change is detected at the other end, attacker can conclude that they are communicating.

Flow watermarking: inject one bit of info (marked or not). Flow fingerprinting: inject multiple bits (e.g. sender IP address) to make the detection of correlation even easier.

This requires a very strong attacker that controls the incoming traffic to a anonymous network (and can observe the outgoing traffic as in the passive case).

Website Fingerprinting Adversary needs only one observation point (ISP, other WiFi user, etc.). By building a database of fingerprints of websites (how many packets are send from each website), the fingerprints can be compared to the observed traffic patterns and the attacker can conclude which websites were accessed. Particularly effective for interactive applications such as health / tax forms.

Traffic Analysis Resistance To resist such traffic analysis attacks, one could combine cover traffic and mixing (as mentioned above). Unfortunately, this introduces a significant overhead and is difficult to scale (large volumes of dummy traffic required). Such methods are only suitable for few applications (e.g. VoIP) with low bandwidth. To be really secure: restrict set of flow duration and bandwidth combination.

7.3.2 Higher-Layer Attacks

Higher-Layer Attacks Instead of observing / comparing / manipulating traffic, an attacker can observe information leaked from the protocols used, e.g. an end-to-end TCP might leak information due to the way the sequence numbers are chosen, how header fields are set, congestion control mechanisms, etc. This either requires packet inspection or being a malicious receiver.

A possible solution is to replace an end-to-end TCP connection with per-hop TCP connections.

HTTP / TLS fingerprints are in most cases unique (OS, browser type, languages, timezone, etc.). See amiunique.org/fp. To protect: either randomize the fingerprint each time a user is accessing the web or make them all look the same (uniformity - as in TOR).

Normally, a de-anonymization attack is done in other ways by, for example, tricking a user into downloading malware / files accessing Internet directly, analysing user behavior (e.g. text written in an online forum). To be anonymous, each layer (even human layer) has to be anonymized since any gap will break it entirely (unlike any other security properties).

7.4 Tor: Second Gen Onion Router

See extra notes.

Additional features:

Tor cells:

Example circuit setup:

Circuit extension:

Hidden services:

Directory authorities:

Censorship resistance:

7.5 Summary

- You cannot be anonymous on your own (anonymity set).
- Anonymous communication is enabled by multiple relays and layered encryption.
- Mix-nets vs. circuit-based systems.
- Anonymous communication can be used for both good and bad purposes.

8 Border Gateway Protocol

8.1 Overview

BGP An exterior gateway protocol designed to exchange routing and reachability information among autonomous systems (AS) on the Internet (update messages over TCP). Routing decisions are made based on paths (list of ASes) and network policies / rule-sets configured by a network admin. For routing within an AS, see the Interior BGP (iBGP).

Path-Vector Routing Protocol A routing protocol that maintains the received path information and updates it dynamically. Each entry of a routing table contains the destination network, the next router and the path to reach the destination.

In BGP, border routers (eBGP peers) send path-vector messages (over TCP) to advertise the reachability of networks. Each router verifies the received advertisements according to its policy.

8.2 Security of BGP

BGP Hijacking BGP routers accept advertised routes from other BGP routers by default (automatic and decentralized routing). This allows for accidental or malicious disruption (blackhole, redirection, interception) by taking over an entire IP prefix(es) ($\approx$ AS) and illegitimately advertise for it. A stronger variation is to originate a more specific (longer) prefix fitting the victim's address space. Traffic will follow the longest matching prefix. Either set up an AS and border router or compromise an already existing router.

Correcting this vulnerability (e.g. crypto keys to verify identity of routers) is technically and economically challenging due to the extend BGP is embedded in the core of the Internet.

Problem 1: BGP does not validate the origin of advertisements.

Problem 2: BGP does not validate the content of advertisements. An AS can easily modify a BGP path (remove an AS to make path look shorter or to attract sources avoiding that specific AS; add an AS to trigger loop detection = DoS or make it look like it has better connections).

BGP Interception Hijacked traffic is captured but still reaches the original / legitimate destination.

BGP poisoning: the hijacked prefix is only announced to and used by some neighbors by triggering loop detection s.t. specific ASes don't accept certain announcements.

BGP communities: can be used to make sure announcements only reach certain ASes, no involuntary learning by peers. Possible by using the *NoExportSelect* action for a specific AS.

Dangers of BGP Hijacking

- Interception / redirection of non-encrypted data (DNS, HTTP, etc.)
- Deriving timing information of encrypted data (fingerprinting).
- Dropped packages (blackholes) and widespread outages.
- Very hard to notice, needs cooperation of ISPs.
- Obtaining fake (TLS) certificates.
- Deanonymizing TOR users.

- Hijacking DNS requests.
- Partition the Bitcoin network.
- etc.

Other Attacks on BGP Most of these attacks are easy to defend against and no longer a big concern.

- Performing a DoS attach by overloading the link between BGP routers which causes packet loss / delay or sending bogus TCP packets that falsely close a session (FIN/RST) or flood a router (SYN).
- Eavesdropping or tampering with messages by tapping the link.

8.3 Countermeasures

Desired Properties We only want ASes that actually own an IP prefix to be allowed to announce it (cryptographic authentication) and routing messages need to be authenticated by all ASes on the path (cryptographic protection, no modification of announcements).

Best Current Practices They are not good enough since they don't address the fundamental problems (who owns an IP address block, is the AS path valid, do data packets actually follow chosen route, etc.)[3]

- Securing the BGP peering session between routers (authentication and priority over other traffic).
- Filtering routes by prefix and AS path.
- Filters to block unexpected control traffic.
- Filtering based on entries in the Internet Routing Registries (IRRs) which contains prefixes.

Solution 1: Origin Authentication Enable issuance of Route Origination Authorizations (ROAs) with a Resource Public-Key Infrastructure (RPKI). They state which AS is authorized to announce certain IP prefixes and determine the maximum length of such.

RPKI: prove ownership of resources by creating a secure database that maps Internet number resources to a trust anchor (e.g. regional Internet registries - RIRs) and utilizing certificates that prove that an AS holds a specific resource. Certificates follow the same delegation as IP addresses from RIRs and are signed / distributed / verified out-of-band (does not require a modification to BGP).

Operation: 1) Receive announcement for prefix v from AS M. 2) Check against ROAs in RPKI for prefix v. 3) Accept if valid ROA for AS M is found, else drop announcement.

Not enough: A malicious AS can append itself on the path with a valid ROA for certain prefix and attract (a fraction of) traffic for legitimate AS.

[3]See MANRS (Mutually Agreed Norms for Routing Security) for current security efforts.

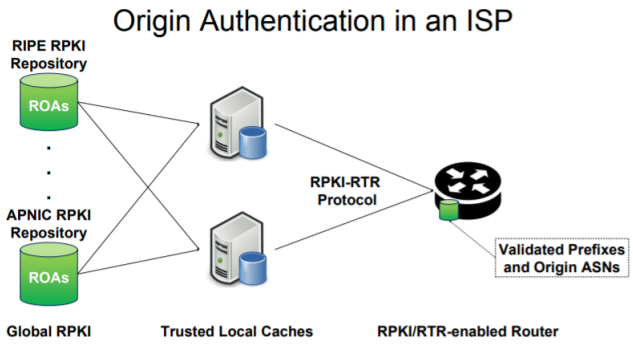

Figure 15: Origin authentication in an ISP.

Solution 2: BGPsec BGPsec secures the AS-PATH attribute in BGP announcements and prevents the crafting of a valid origin by prepending ASes along with path poisoning. It's basically origin authentication + cryptographic signatures. Received messages are signed to prove that a path was correctly updated (next AS is included in the signature).

BGPsec can validate that the AS path indicates the actual order of traversed ASes and disables modifications (adding/removing an AS). RPKI is used to verify AS key material (similar to solution 1).

BGPsec is in incremental deployment which allows for protocol downgrade attacks (if security is not prioritized when dealing with non-BGPsec BGP routers). It is challenging to figure out how to effectively prioritize security (over business relationships). Furthermore, performance is degraded since BGPsec does not allow for prefix aggregation and involves complex crypto (slower convergence).

Challenges for large-scale deployment include: different message formats, complete and accurate registries and PKI. In general, upgrading equipment is expensive, cooperation amongst many entities is required and the benefits are unclear for first-movers.

Extensive Monitoring Another approach to problem 2. BGP update messages are monitored and compared to past history. Remember which AS originates which prefix and AS-level edges / paths. Delay adoption of unfamiliar routes if possible.

This can be realised with an out-of-band detection mechanism (e.g. *ARTEMIS*, *Prefix Hijack Alert System*, etc.).

SCION Proposal to redesign inter-domain routing since BGP was not created with security in mind. Introduces trusted entities, enables simple security patches, etc. Other approaches include Named-Data Networking / Information-Centric Networking, Accountable Internet Protocol, Passport: Secure and Adoptable Source Authentication, etc. See later Section for more details.

9 Real World Network Security

Guest lecture.

10 (Distributed) Denial of Service Attacks

Denial of Service Attack Making a service or network resource unavailable to its intended / legitimate users. Typically achieved by exhausting available resources by sending an excessive amount of traffic.

To mitigate such attacks, it is common to use a combination of various mitigation techniques at different points in the network hierarchy.

First, an attack has to be detected (by comparing it to normal / human traffic patterns) and then it has to be filtered.

Distributed DoS Attack Many different sources simultaneously (often with botnets). Often used to extort companies. Harder to track and take down than a regular DoS attack and also easier to hide identity. Allows for a virtually unlimited bandwidth for flood attacks.

Attack Targets See Figure 16 for an overview.

	Network links	Network devices / networking stack	Applications
Description	Volumetric attack	Protocol attack	Application-layer attack
Unit of measurement	Bits per second (bps)	Packets per second (pps)	Requests per second (rps)
Used mechanisms / examples	• Reflection and amplification • Shrew attack	• Reflection • State exhaustion • SYN/ACK floods • Fragmentation	• Computational complexity • Hash collisions • Slowloris
Defenses	• Filtering, traffic scrubbing • Black-hole filtering	• Cookies • Rate-limiting	• Randomized/keyed hash functions

Figure 16: Targets of a (D)DoS attack.

10.1 General DoS Attack Techniques

Features Facilitating DoS Attack

- Attacker controls significantly more resources than victim.
- Attacker needs to expend significantly less resources than victim.
- Attacker can hide his identity / continually change it.
- Victim needs to expend a significant amount of resources before being able to assess the legitimacy of requests.
- Attacker can instruct / trick other entities to send traffic on their behalf.

Availability in a public Internet is threatened by different types of DoS attacks

Link-flooding attacks	Attacker floods network links with excessive amount of traffic
	Can target **access links** (last mile) or **core links** in the network
	Often executed using botnets and/or amplification techniques
End-system attacks	Attacker exhausts **computational** or **memory resources** of victim
	Often possible due to other defense mechanisms such as firewalls
	Examples: **state exhaustion**, **signature flooding**
Control-plane attacks	Attacker disrupts important **control-plane mechanisms** or **access to services**
	Services are essential for a functioning network
	Examples in SCION: beacon server, path server, certificate server

Figure 17: Different kinds of DoS attacks.

(IoT) Botnets A (large) set of compromised machines (often IoT devices) connected to the Internet. They execute malicious code and can be controlled via command and control (C&C) systems. They are often geographically distributed.

Components: command and control infrastructure to push commands to bots (often owned by attackers), zombies = devices under control of the attackers (often vulnerable hosts infected by malware) and bots - software that contains the malicious business logic.

To spread, an infected device will scan and look for open ports (e.g. sending TCP-SYN packets to randomly generated IP addresses), brute force a login, report target to controller and infect by connecting to found device and uploading malware.

IoT devices since you can have many with uniform configuration (configure-and-forget), are poorly secured (e.g. default credentials), not regularly updated (in regards to security) and often connected to the Internet without bandwidth limitations.

E.g.: Mirai botnet (mostly vulnerable webcams).

Mitigations: patching (automatic security updates for the device's full lifetime), credentials (not hardcoded and forced to change default passwords), monitoring (ISPs should actively monitor network for suspicious traffic).

Reflection and Amplification Need publicly accessible servers and the ability to spoof source address. Ideally, the responses caused should be (much) larger than the requests (amplification). Can be combined with a botnet consisting of master and agent nodes to exponentially grow volume.

Typical reflectors are: DNS (max. amplification 180), NTP (max. amplification 500) and Memcached (max. amplification 50'000) [4].

An example DNS query that triggers a big response is the ANY query - it returns all DNS records of a domain (including the lengths SOA records). Nowadays, most DNS resolvers / authoritative servers refuse to respond to ANY queries or limit the response size.

Amplification factor: response bytes / request bytes.

How to:

- Choose open service as reflector (e.g. open DNS resolver).
- Craft a request that triggers (much) larger response.

[4] Currently, NTP vulnerability is closed along with Memcached having UDP disabled.

- Send packet where source address is set to victim's address.
- Reflector sends reply to victim.

Mitigations: perform access control, implement response rate limiting (RRL) and ensure small amplification factors (ideally < 1)[5].

Address Spoofing Source address in an IP header can be set by the sender and in a connectionless protocol, such as UDP, a server cannot confirm the actual sender.

Defenses: Address filtering by ISPs (hosts should use their own addresses - would need to be globally deployed and the incentives are poor since only other ISPs profit), using connection-based protocols (such as TCP - comes with additional latency and other ways to DoS (state exhaustion)) and cryptographic source authentication (again, other attack possibilities if asymmetric crypto and requires symmetric key distribution / PKIs).

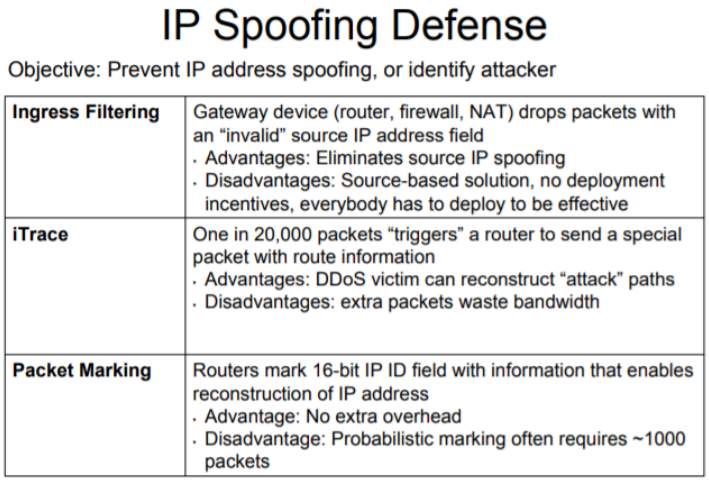

Figure 18: IP spoofing defenses.

10.2 Specific Attack Examples

Volumetric: Shrew Attack Achieving the same effect of bandwidth-based DoS (high-rate attack traffic) with low-rate attack traffic by exploiting the TCP congestion control feature (exponential backoff if packet loss is detected).

Periodically sending short bursts to the target link / router denies the bandwidth of legitimate TCP flows at it makes TCP believe there is a long-term congestion. The bursts are most commonly send exactly when client is trying to send legitimate traffic (MITM attack in this case since the attack needs to be coordinated).

Temporal lensing: concentrating a low-bandwidth flood into a short, high-bandwidth pulse s.t. even a low-bandwidth attacker can perform a shrew attack. By knowing the attack path latencies, e.g. from reflector to victim, the attacker can send packets at different times to reflectors with varying path latencies s.t. overall traffic arrives at victim at the same time. With lensing, attackers can achieve peak bandwidths larger than their actual upload bandwidth. Lensing can be combined with amplification to create even larger pulses.

[5]WireGuard ensures that the responder's first message is smaller than the initiator's.

Volumetric: Coremelt Adversary controls botnet distributed across Internet. Bots send traffic between each other (not to victim, desired traffic). Adversary exhausts bandwidth on victim link in a per-flow fair sharing system.

Volumetric: Crossfire Adversary controls botnet distributed across Internet. Since route optimization leads to few links actually being used to target a region to rest of the Internet, adversary can contact selected servers to overload these links. Can disconnect target region from remainder of Internet.

Protocol: DNS Flooding / NXDOMAIN Attack Overwhelm the victim's authoritative name servers by querying many non-existent subdomains of victim domain. The DNS resolvers query all authoritative name servers in turn and thus a name server will not reply to legitimate requests. The more attackers and resolvers the better.

Protocol: Session State Exhaustion Two-way communication channels are identified with unique session numbers (= session state). Numbers are known at a server to match subsequent requests to right session. By exhausting the session state table of the server, it can no longer accept new connections, existing connections are dropped and maybe the server / service even crashes.

Generic mitigations: Encode state in a unique but determined way that allows server to validate state in client reply. No state at server is needed (other than salt). Ensure the encoding cannot be tampered (use crypto-hashes, unique data known to server only and changes over time, etc.), e.g. $B =$ Hash(salt, A).

SYN flood attack: TCP three-way handshake first message is client sending a SYN packet with a random sequence number A and server keeps $A+1$ and B (random seq. number of server SYN+ACK). With spoofed source addresses, a server can be flooded with SYN messages and the state table will eventually overflow.

SYN flood attack mitigation: SYN cookies - no state tables needed if initial TCP sequence number is particularly chosen (e.g. $B = F$(time, IP address, port, ...). Apply F after ACK of client to validate and establish connection.

Application-Layer: Algorithmic Complexity Induce the worst-case behavior in a vulnerable algorithm. The larger the difference between worst and average case, the more vulnerable it is. See Figure 19 for a comparison.

Hash table lookup collisions: collisions are common due to small size. Resolutions include chaining, open addressing, etc. An attacker can intentionally pick a bad input sequence that causes worst-case hash table collisions. Countermeasures include universal hashing (guaranteed zero to little collisions for any input), hash randomization (hard to find bad input), etc.

Regular Expression DoS (ReDoS): provide a regular expression that takes a very long time to evaluate (matching). Most regex implementations have exponential time worst case complexity - meaning that the time taken can grow exponentially in relation to input size.

Algorithm Name	Best Case	Worst Case
Optimized Insertion Sort	$\Theta(n)$	$\Theta(n^2)$
Quick Sort	$\Theta(n \log n)$	$\Theta(n^2)$
Optimized Quick Sort	$\Theta(n \log n)$	$\Theta(n^2)$
3-way Quick Sort	$\Theta(n \log n)$	$\Theta(n^2)$
Sequential Search	$\Theta(1)$	$\Theta(n)$
Binary Search	$\Theta(1)$	$\Theta(\log n)$
Binary Search Tree Lookup	$\Theta(1)$	$\Theta(n)$
Red-Black Tree Lookup	$\Theta(1)$	$\Theta(\log n)$
Separate Chain Hash Lookup	$\Theta(1)$	$\Theta(n)$
Linear Probing Hash Lookup	$\Theta(1)$	$\Theta(n)$
NFA Regex Match	$\Theta(m+n)$	$\Theta(mn)$
Booyer-Moore Substring	$\Theta(m+n)$	$\Theta(mn)$
Prim Minimum Spanning Tree	$\Theta(V+E)$	$\Theta(E \log V)$
Bellman-Ford Shortest Path	$\Theta(1)$	$\Theta(V(V+E))$
Dijkstra Shortest Path	$\Theta(1)$	$\Theta(E \log V)$
Alternating Path Bipartite	$\Theta(V)$	$\Theta(V(V+E))$
Hopcroft-Karp Bipartite	$\Theta(V)$	$\Theta(E\sqrt{V})$

Figure 19: Comparison of best and worst case complexities.

Application-Layer: Slowloris A single machine can take down another machine's web server with minimal bandwidth by:

- Keeping many connections to target web server open and holding them open as long as possible.
- Opening connections and sending partial requests.
- Periodically sending subsequent HTTP headers adding to the request but never completing it.

Affected servers will keep these connections open, filling their maximum concurrent connection pool and eventually denying additional ones.

Mitigations: increase max. possible connections to server, limiting connections from single IP address source, require minimum transfer speed per connection, restrict length of time client is allowed to stay connected. Or: set up reverse proxies, firewalls, load balancers or content switches.

10.3 Countermeasures

- **Ingress Filtering:** Remove packets with illegitimate source IPs.
- **Computational Puzzles:** Slows down attacks, achieves per-computation fairness.
- **Cloud- or ISP-based Filtering:** Delegates defense to cloud / ISP.
- **Network Capabilities:** Allows victims to block unwanted traffic closer to the source.
- **IP Traceback:** Reveals real source IP of packets.

Guarantee Little Downtime

- **Redundancy:** No single point of failure, $N + 2$ systems running under normal operation (if two systems fail, service still available), > 2 geographically diverse connections / independent Internet connections.
- **Monitoring and Rapid Detection / Automatic Failover and Eviction:** Long term monitoring to assess periodicity and peak periods / loads. Over provision such that resources cover majority of extreme peak loads (average can be misleading).

- **Failure Resiliency:** System can tolerate various temporary component failures and gracefully degrades upon too many.

Cloud-Based DDoS Mitigation Service Traffic can be redirected to a cloud provider for filtering or content delivery by changing BGP / DNS (scrubbing / rerouting). Some provide CDN service to serve requests from many locations.

Can be easily bypassed if the victim's IP is exposed since most clouds use DNS (BGP-based is harder to bypass). Also prone to privacy violation. High cost and requires continuous subscription.

In-Network- / ISP-Based DDoS Mitigation Service ISP redirects traffic to high-capacity scrubbing center and sends back filtered traffic to destination (always or on demand). Scrubbing center uses deep packet inspection (DPI) and connection patterns to filter malicious traffic.

Remotely Triggered Black Hole (RTBH) Filtering Can be used to mitigate volumetric attacks. Install rules to drop traffic based on source or destination addresses in border routers of an AS (= black hole). Can be achieved through BGP updates (most likely triggered by manual intervention).

10.4 SCION

One way to completely redesign the internet. Achieving global communication guarantees on the public internet (high security, formal verification from the start). High efficiency due to path-aware networking - sender knows the path and can select among options - and multi-path communication (= path optimization and load balancing). Additional principles include:

- Stateless packet forwarding (routers keep almost no state, no risk for inconsistent forwarding state)
- Instant convergence routing (all paths are instantaneously usable)
- Sovereignty and transparency for trust roots
- Per-packet authentication and verification possible on routers
- Formal verification of protocols and code
- Immune against routing attacks (e.g. BGP prefix hijacking)

10.4.1 SCION Overview

Isolation Domain (ISD) A grouping of autonomous systems. An ISD has a core consisting of ASes that manage it and provide global connectivity (upstream = reach core). Each ISD has a root of trust governing it (trust root configuration, TRC) and can thus function autonomously without external trust.

Figure 20: SCION goals / motivation.

Control Plane Responsible for the routing mechanisms. In SCION, the control plane constructs (path exploration) and disseminates (upon sender request) path segments. Each AS contains border routers, beacon servers and path servers. The control plane is also responsible for the dissemination and renewal of certificates needed to verify segments.

Intra-ISD Path Exploration: Done with beaconing. The core ASes initiate path-segment construction beacons (PCBs) which are flooded throughout the entire ISD - downstream ASes learn how to reach core ASes through the paths that the PCBs have taken (on the way down, each AS that sees the PCB appends itself as a next hop, additional to a MAC and its signature).

Inter-ISD Path Exploration: Same concept, just a lot denser than in the intra case. PCBs are distributed only among core ASes to create core-path segments.

Path Server Infrastructure: Path servers offer a lookup service (similar to DNS) to find down-path, up-path and core-path segments. The core ASes of each ISD operate the core-path and down-path server infrastructure (consistent and replicated) - selected path segments are chosen, stored and announced globally. Each non-core AS runs local path servers that serve selected up-path segments to local clients (no global announcements) along with resolving and caching responses of remote AS lookups.

In total, a host fetches the local up-path segments, global down-path segments and core-path segments if needed (to connect up- and down-path segments) by contacting its local path server with (ISD, AS) which might contact more path servers if info is not cached - this created an end-to-end path (which might be shorter than all segments combined - see yellow paths in Figure 21).

Path Combinations: Not all path combinations are feasible due to security reasons. See Figure 22 for all possible combinations.

Data Plane Responsible for the actual packet forwarding. Path segments are combined to form a path. Segments (plus some metadata like geographical location or link latency) are included in the packet header (encrypted), therefore a sender knows exactly which ASes are traversed. A router only verifies the authenticity of the information (two AES operations replace longest-prefix match).

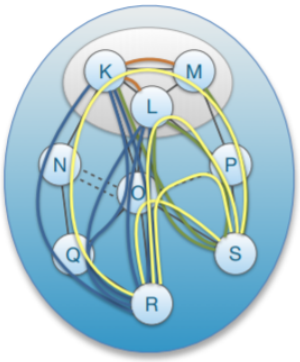

Figure 21: Possible end-to-end paths from AS R to AS S (in yellow).

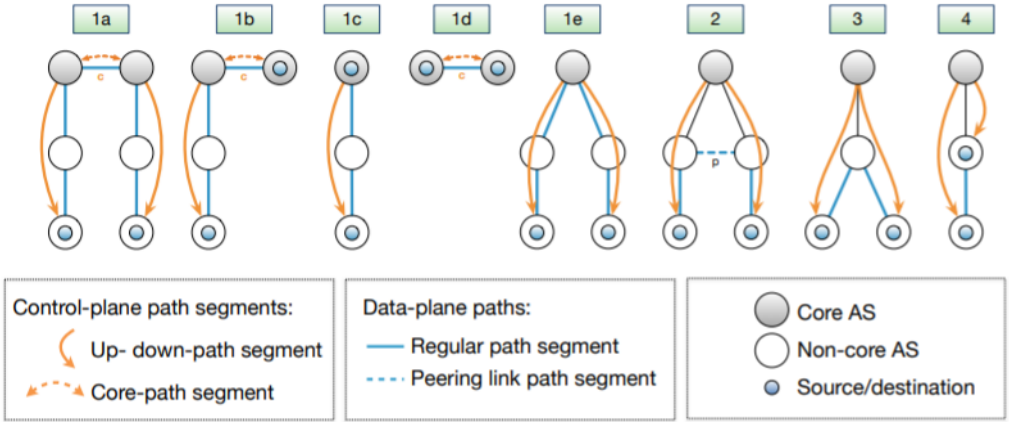

Figure 22: All possible ways to combine path segments.

SCION Drawbacks

- Initial latency inflation to obtain paths during lookup (amortized by caching and path reuse since most destinations are reused).
- Bandwidth overhead due to paths in the packet headers - around 80 additional bytes (but enables path control and a simpler data plane).
- Increased complexity in key management due to new certificates like TRC certificates (necessary for a high security design).
- Initial set-up cost due to training network operators and installing new infrastructure (methods exist to facilitate deployment and new properties become possible with it).

SCION Deployment Since the AS infrastructure inside an ISP is being reused, there is not a lot of change necessary in the internal network infrastructure. We just need to set up core routers at the borders of an ISP (peering with other SCION-enabled networks and collect customer accesses). For end domains, we just need a SCION IP gateway (SIG) that enables seamless integration of SCION capabilities in end-domain networks. SIGs are then connected to SCION (border) routers. No upgrades of end hosts / applications are needed, it's just an additional packet header.

Incremental deployment is possible. Important to not rely on BGP for inter-domain operation s.t. we don't inherit its vulnerabilities. For local communication, intra-domain network architecture can be reused (intra-domain routing).

10.4.2 SCION Security Insights

Global Communication Guarantees Especially in the presence of adversaries. Goal: if a (routing policy compliant) path of good ASes exist, a sender can find, use and achieve minimum bandwidth guaranteed on that path. Challenges include: network routing instabilities / misconfigurations and DoS attacks are various levels (control and data plance, end hosts, etc.).

Stable Forwarding and Multi-Path: for the first challenge, these two concepts are necessary. Single-path forwarding cannot achieve strong availability guarantees (convergence, equipment failure, packet loss - path available / connectivity but shitty and needs manual intervention, etc.). With stable forwarding we have packet-carried forwarding state that protects forwarding from routing instabilities. With multi-path, we ensure the presence of several paths and as long as one works, end-to-end connectivity is assured. Overhead is low with the path segment combination method (instead of full paths). But, secure routing is insufficient in the case of outages caused by bottleneck links or continuing announcement of failed / congested routes since announcements in these cases are still legitimate.

DoS Attacks: SCION has several components that guarantee strong security and high availability (see Figure 23).

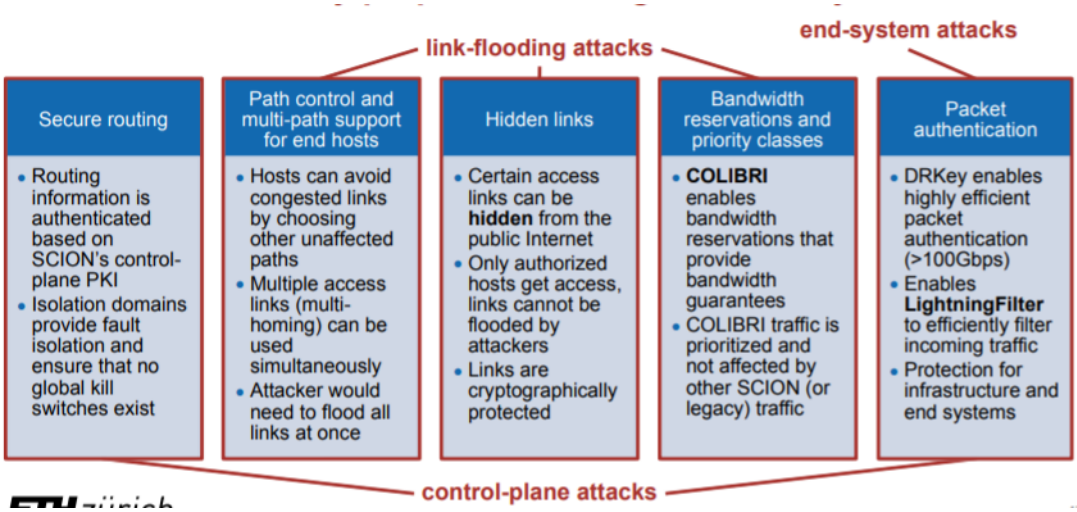

Figure 23: SCION protecting against DoS type attacks.

High-Speed Packet Processing If you do it right, cryptographic operations can be performed on a per-packet basis. E.g. with a 400 Gbps link, a 64-byte packet arrives every 1.3ns. Symmetric crypto enables high-speed processing through parallel processing and pipelining - e.g. taking only around 100μs per signature (compared to a DRAM memory lookup that takes around 70ns). Not possible with asymmetric crypto.

SCION offers a global framework for authentication and key establishment for secure network operations. DRKey = dynamically recreatable keys - creating crypto keys on the fly (within 20ns). With DRKey and EPIC, per-packet source authentication in software possible under 100ns. Control-plane PKI means that every AS has a public-key certificate that enables AS authentication and for each ISD to operate in a sovereign manner.

- **DRKey:** use a per-AS secret value to derive keys with an efficient pseudo-random function (AES operation). With the secret value, a key created for another AS can be derived (fast) but without it (other AS) the key has to be fetched by contacting a local key server.

- **Key Server Infrastructure:** each AS has a key server building the backbone of the key hierarchy. Servers are responsible for key exchange, management and local key establishment. After establishing AS-level keys, symmetric keys for end hosts can be provided and used to provide source authenticity of packets (without a costly key exchange). Each host is required to contact their local key server. See notes for a better explanation.
- **LightningFilter:** SCION firewall that authenticates packets at very low overhead (no FP/FN, no heuristics). Sender locally fetches remote LightningFilter's DRKey and the filter then authenticates packets by deriving the DRKey. Normal to combine it with normal firewall (LF at border and firewall closer to destination). Also allows for history-based filtering by keeping track of rate of key requests of AS (and blocks if they overshoot). And duplicate suppression with bloom filters.
- **Every Packet Is Checked (EPIC):** per-packet source authentication by every router and destination - possible with DRKey and a per-packet unique hop field (no duplication possible). Assumes global time synchronization. Same as in LF, source needs to first fetch all the keys to compute per-hop fields. Prevents malicious router replays / amplification and MAC bruteforce.

COLIBRI Provides practical and scalable global QoS. Stable paths that ensure reservations are not affected by routing changes. Multi-path system to search for paths with sufficient bandwidth. No per-flow state on routers (enables scalability) - possible with DRKey (per-packet source authentication), probabilistic large flow detection to detect overuse, per-flow stateful control-plane (server infra). Per-neighbor fairness (admission decision, ISP configs).

Admission Algo with Per-Neighbor Fairness: instead of per-flow fairness (normal Internet), we have per-neighbor fairness. Each AS defines N2N minimum bandwidth guarantees (that can be computed for any path). Algo guarantees that no set of ASes can reserve a disproportionate amoung of bandwidth through any link.

Bandwidth Reservation Simplifies transport layer due to no need for sophisticated congestion control (simply use constant bitrate - good for streaming traffic). Low loss rate - low amount of ACKs. Enforce fairness at level of admissions. Traffic engineering.

11 Firewall Intrusion Detection and Evasion

11.1 Overview

Firewall A system protecting or separating a trusted network from an untrusted one. It either permits, denies or proxies data through a network with different levels of trust. = Access control policy between two networks.

This is not enough. Many ports need to be kept open for legitimate applications to run. Evolution messes with legacy firewall technology (P2P. instant messaging, Web 2.0 - multiple functions for single-function port, etc.).

- **Network firewall:** Software appliance running on specific hardware or as a virtual instance. Filters traffic between two or more networks. Protects different network segments.
- **Host firewall:** Layer of software on a host that controls traffic going in and out of a single machine. Protects single host.
- **Stateless firewall:** Examine a packet at the network layer and base decision on packet header information. Application independent and good performance / scalability, but no context.
- **Stateful firewall:** Also keep track of state of network connections and base decision on packet header and session state. More powerful rules are possible but state can be inconsistent / nonexistent (UDP) and explode ((D)DoS).

Filtering Rules

- **Ingress:** Incoming traffic, low to high security network.
- **Egress:** Outgoing traffic, high to low security network.
- **Default:** Define what to do if no rule matches (accept / reject).
- **Deny Access:** Either silently drop or reject (*ICMP Destination Unreachable*) packet.

Next Generation Firewall (NGFW) Performs deep packet (payload) inspection and takes application and protocol state into account (on top of typical firewall functions). Allows for powerful rules and application / protocol awareness. Cons: needs to support many application protocols, impacts performance / scalability, deals with inconsistent state (host vs. firewall), etc.

Web Application Firewall (WAF) Protects web-based applications from malicious requests by filtering based on requests / signatures (SQL injection, cross-site scripting, buffer overflow, checking number of form parameters, etc.). Allows for static or dynamic black- / whitelisting. Con: false positive problem.

WAFs are often implemented as a reverse proxy (client outside of internal network) to protect public facing web applications.

Deployment Challenges How to protect a large number of hosts / endpoints / network segments.

- **Complexity:** Rules can be complex, thousands of rules per firewall are common, detection between channels is out of sync (mail vs. net vs. proxy).
- **Management:** How to change / remove rules? Who has the permission? What tools?
- **Incentives:** Infrastructure team is paid for providing connectivity and blamed for disruptions and security team is paid to protect and disrupt connectivity...

11.2 Firewall Attack Methods

IP Source Spoofing Spoofing the source IP address. Works for stateless protocols (e.g. UDP, ineffective for TCP).

Artificial Fragmentation A fragmented packet might pass a firewall if it's not able to properly reassemble it (e.g. because they're out of sequence). For example, the port number is only in the first fragment. Also works by splitting the attack into multiple packets or overlapping the individual fragments.

Denial of Service Firewall state explosion. Need to define a good fallback policy.

Tunneling / Covert Channels Creating a capability to transfer information between processes that are not supposed to communicate. For example, putting data in ICMP ping packets or using DNS requests as a channel. Furthermore, attacks can also be performed through a VPN tunnel.

Payload Encodings Use different encodings allowed on the protocol / application in use and addition of noise to confuse detection. Different encoding and obfuscation schemes can be combined with noise insertion in millions of ways and encodings are not necessarily unique. Furthermore, undefined or border cases are very effective for detection evasion. There might also be different implementations of decoding on target application vs. detection engine decoding.

E.g.: URL encodings (% encoding, path character tranformations) or HTML obfuscations (UTF-7 / UTF-16 / UTF-32 character set encoding, chunked encodings, different compression schemas, Base64 encoding, etc.).

Firewall Vulnerabilities There are several vulnerabilities found in firewall software from security vendors. Firewalls are complex! See examples in Figure 25.

11.3 Intrusion Detection Methods

Key Features of Intrusion / Attack Detection

- **Reactive:** System can only detect already known attacks.
- **Proactive:** System can detect known and new, yet unknown attacks.
- **Deterministic:** System always performs the same given the same input. Reason for alert is known.

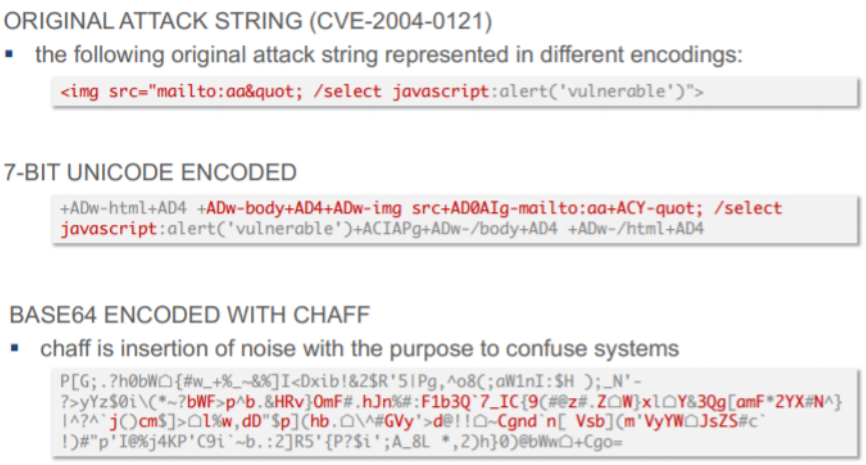

Figure 24: Attack string in different encodings.

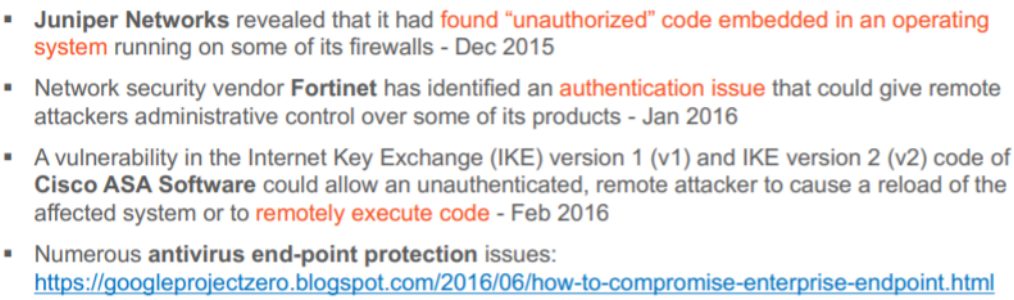

Figure 25: Some vulnerabilities found in commercial firewalls.

- **Non-Deterministic:** Detection is fuzzy (heuristics, ML, sandboxing, etc.) and depends on state. Reason for alert is typically not known.

Detection Techniques See basic detection approaches in Figure 26.

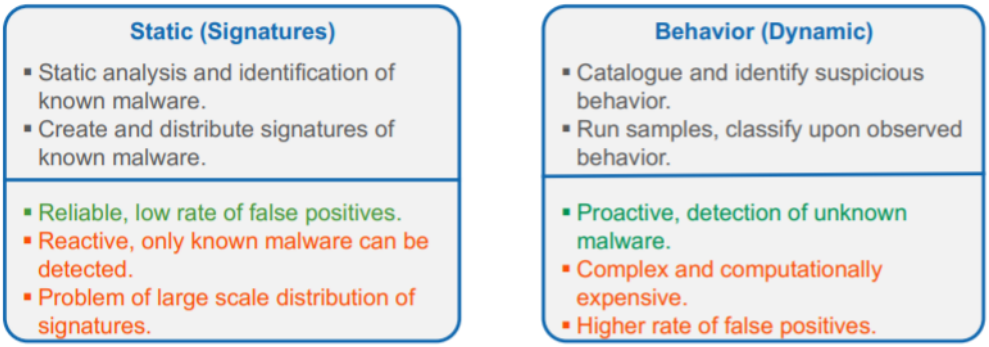

Figure 26: Basic approaches to intrusion detection.

Key features include:

- **Protocol Analysis:** Analysis and decoding of protocols. Reassembly and normalization of traffic.
- **Signatures:** Compare attributes of observables to black- / whitelists or to patterns of known

attacks / exploits / malware. A signature can be simple groups of checksums / byte arrays or complex hundred lines of code.

- **Sandboxing / Behavior:** Suspicious file is executed within a virtual environment. Specific actions are categorized and labeled as good / bad.
- **Machine Learning:** Recognize complex patterns and make decisions based on data / assumptions formed from previous data. Learn (good vs. bad), extract features, train and test.
- **AI:** not just learning how to solve a specific task very well (ML) but actually learning how to improve itself to solve new previously unknown tasks. Goal: increase chance of success, not accuracy.

Signature based approaches will stay and be accompanied by more sophisticated unsupervised ML approaches.

Signature Based Detection Core concepts still lie at the heart of all modern detection systems and will continue to be integral for the foreseeable future. Reactive and deterministic. Progression / Sophistication of such systems depend on human signature writers. Bad with mutations / polymorphism. Characteristics:

- Able to promptly identify and label a threat.
- Different signature systems are used together for more accuracy.
- Unique signatures or signature artifacts are created for new threats.
- Signature databases / online lookups are frequently updates.

Indicators of Compromise (IoC) similar, not a full signature (mostly IPs or file hashes). E.g. Yara Rule.

One-Dimensional: common in all systems, fastest and most efficient way of categorizing a data artifact. Simply a boolean output (good / bad). Doesn't need many resources, is fast and has a low number of false positives but is reactive, needs frequent updates to signatures and uses humans.

Two-Dimensional: regular expression functions and string matching, fundamental for anti-malware / IDS / DLP systems, easily capable of identifying previously known exploits and host enumeration techniques. Low / medium resource requirements, more flexibility with patterns and low false positive rate. Cons: same as above.

Multi-Dimensional: instead of triggering on a single signature, a multi-dimensional signature was created, can label actions as suspicious or bad (sandboxing / network behavioral monitoring). Classify and label threat after a certain threshold of good or bad activities is met. Is partially proactive and more efficient / effective than above approaches.

Sandboxing Based Detection Basically running malware in a detonation chamber by examining / monitoring runtime behaviour of a sample and comparing behavior against a list / rules previously developed in a lab or via machine learning (behavior classification - no humans!). Is proactive and mostly deterministic. Doesn't require updates to signatures. Cons: resource intensive, high latency and difficult to scale.

Limitations of Intrusion Detection

- Unable to inspect encrypted traffic.
- High number of false positives.
- Packet capturing and analysis at high link speed (tens of Gbits/s).
- Minimize latency introduced by inspection engine.
- Application level attacks (JavaScript, etc.).
- Policy / signature management.

Accuracy vs. Precision For decision making (block/pass) the consistency of accuracy is most important. Accurate detection is very challenging (balance zero FN and zero FP).

Accuracy: how close is the measured value to target value (bias).

Precision: Values of repeated measurements are clustered and have little scatter. Can be far away from target value (variance).

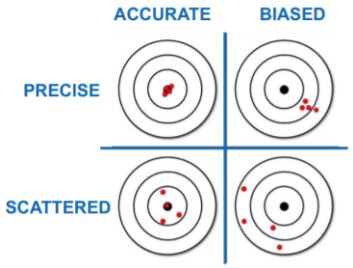

Figure 27: Accuracy vs. precision.

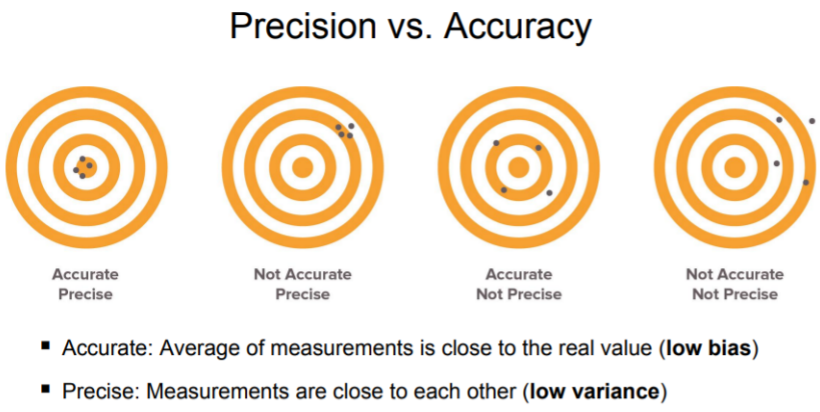

Figure 28: Another accuracy vs. precision.

Sensitivity vs. Specificity Sensitivity measures the proportion of positives that are correctly identified (TP / all positives resp. FN + TP). Specificity measures the proportion of negatives that are correctly identified (TN / all negatives resp. FP + TN).

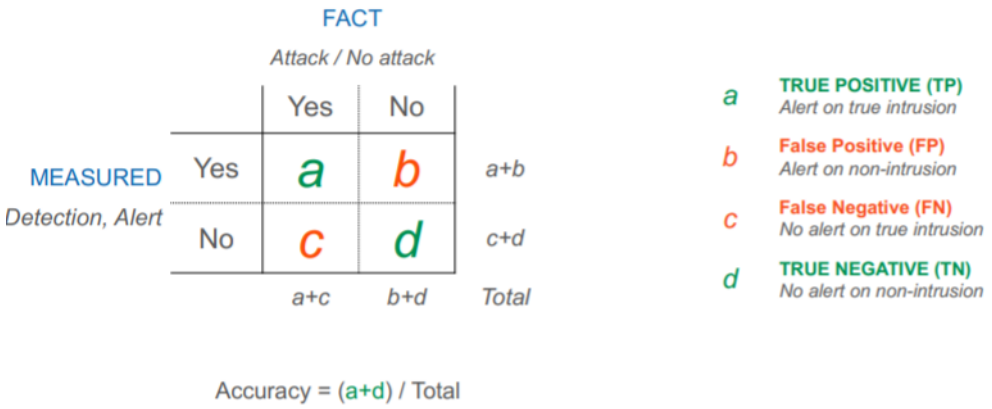

Figure 29: Accuracy and type of measurements.

Detection Performance

- Accurate detection is very challenging when rate of attacks is very low.
- Different detection techniques achieve different sensitivity and specificity.
- Use a combination of diverse tests to increase precision.
- Context is important (host vs. network based detection).

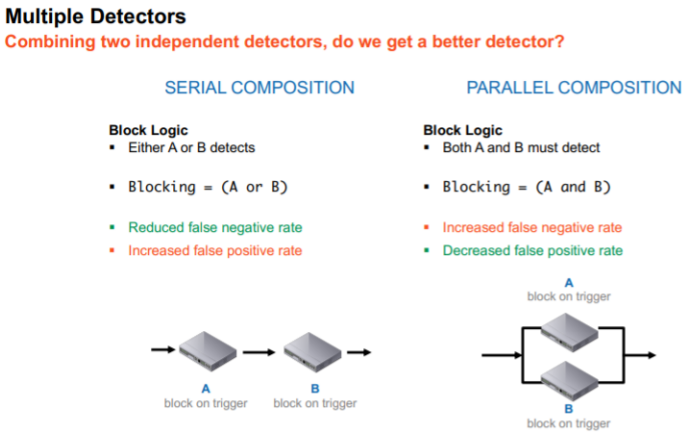

Figure 30: Multiple detectors.

11.4 Layered Security - Filtering and Protection

11.5 Detection Evasion by Design

Step 1: Create Initial Malware Create malware core functionality, such as key logger, web-injection, attacks, spreading, communication, etc. by hiring a coder, buying / stealing it or do it yourself.

Step 2: Crypter and Serial Variants Encrypt it s.t. signature detection systems and static analysis processes are ineffective. Use different keys to create serial variants / permutations. Encrypt contents of malware executable. Upon execution, only decrypt sections of code on the victim's computer.

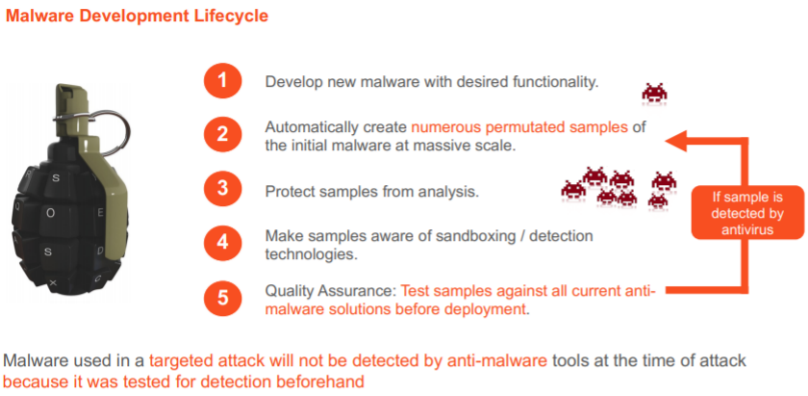

Figure 31: Malware development life cycle.

Step 3 and 4: Protection Against Debugging Use protector technology that detects the use of debuggers or virtualization techniques. If seen, a malware can do different operations (hiding malicious intention or trigger exploitation of virtual environment).

Polymorphism techniques: How to mutate code while keeping the original algorithm intact. Done by manipulating structure of source code of malware by reordering and replacing common programmatic routines. Swapping of equivalent code constructs, changing order of code (registers, instructions, function definitions), inserting noise (redundant code, non-occurring exceptions, functions that are not called), compiler modulation (different compilers can result in different binary code output), etc.

Step 5: Quality Assurance Pass malware through many commercial antivirus products (automated). Only use services that do not submit malware samples to vendor!

12 Malware Analysis and Prevention

Guest lecture.

12.1 Threat Analysis

What to do with a Potential Threat After finding out what to analyze, do automated testing to get a first impression - we may already know it. One can use internal sandboxes or use scanners. However, many malware samples try to prevent automated analysis (can detect a VM). Use a hex editor to find out if it's a portable executable (e.g. .exe) file. Such files have a distinct and well-defined structure.

Analysis generates a lot of metadata that one can further analyze and put into a database for clustering (e.g. does a website host a lot of malicious URLs?).

Blackboxing (Dynamic Analysis) Executing a sample on a dedicated system (virtual or real) to see what it does (not interested in how, we just wanna know what kind of malware it is). Monitor all system API calls, analyze logged information, dump decrypted content from memory (if malware is originally encrypted), etc.

Blackboxing can't tell you everything! Use Whiteboxing as well.

Whiteboxing (Static Analysis) Actually analyzing and reading the code by using a debugger or a disassembler. Can be helpful to find hidden functions (e.g. just doing malicious things on a specific day/condition, etc.).

12.2 Prevention Methods

A detection rate without knowing the false positive rate is useless. You could just classify everything as malicious to catch all malware (high false positive rate).

Also, see Firewall chapter.

Main Types

- **Pattern Matching:** signatures, static ML, data loss prevention, etc. - what is it?
- **Analysing Behavior:** anomaly detection, post-execution ML, web application firewall, etc. - what does it do?
- **Prevent Unwanted Access/Changes:** hardening, firewall, app isolation, etc.

Some Other Protection Concepts

- **Deception:** honeypot an attacker.
- **User Entity Behavior Analysis:** zero trust.
- **Endpoint Detect and Response (EDR / XDR):**
- **Network-Based Detection and ISP:** anomaly detection in network, etc.

13 Internet of Things

Safety vs. Security Safety is the protection against random, unwanted incidents - resulting from coincidences or driven by the environment (which does not adapt to bypass safety measures). Security is the protection against intended incidents - resulting from a deliberate planned act and driven by targeted attacker.

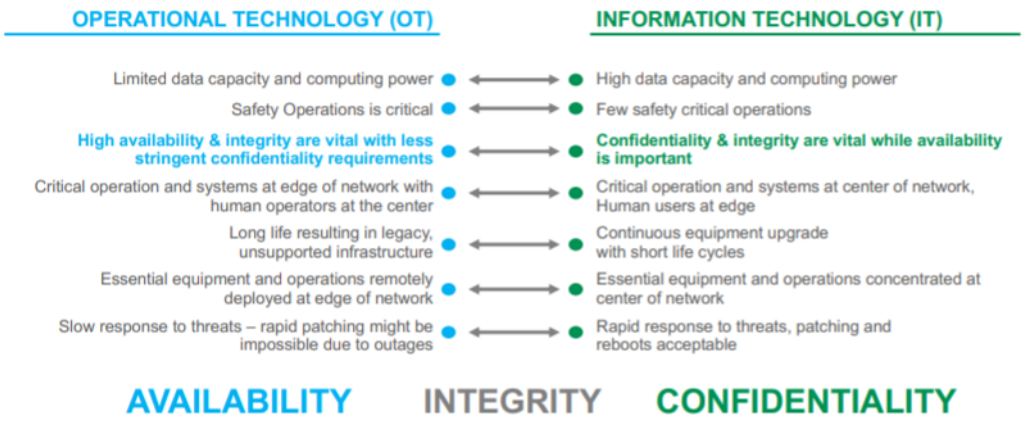

Figure 32: Key differences between operational and information technology.

Attack Surface IoT connects innumerable everyday devices and systems. Devices can have insecure software / vulnerabilities and insecure defaults, cheap components, be lacking of an update mechanism. Furthermore, since they're replicated, finding a fault in one can give access to all (no unique identity). Backend service is the central control and can be attacked in regards to data (breaches, privacy, etc.). Both ends are connected with by an insecure communication channel (weak or no crypto, lack of authentication, etc.). Now, closed systems are opened up to remote access and control!

IoT devices are perceived to have very low risk in comparison to a computer. An attacker can exploit this risk perception (easy targets).

IoT devices are mainly designed to have high availability with safety but no security. They need continued security maintenance. Also, certification timeline is often outpaced by cyber security (patches make a certification unusable).

Challenges	What is needed
The security of individual components (e.g. technologies) does not imply the security of the complete system (connected vehicle ecosystem).	▪ Design systems with redundancy and resiliency (fail safe, fail secure) ▪ Realistic testing plans for the complete system (end-to-end)
The continued innovation of attackers, threats, technologies, society, and use-cases creates a dynamic and adaptive threat landscape.	▪ Prepare for continued adaptation to new threats, no matter what the driver or domain behind the attacker or threat. ▪ Comprehensive and continued security monitoring
Software drives everything, and there is no such thing as secure software. Prepare for the continued discovery and publication of vulnerabilities in software and hardware.	▪ Active management of vulnerabilities (coordinated disclosure, bug bounty) ▪ Robust and scalable process to deploy security updates timely and efficiently – on any connected device
We depend on a complex supply chain of hardware and software components, which can not be fully controlled. Assume that some components are already compromised.	▪ Systematic security and integrity testing of all critical components (reverse engineering of software & hardware). ▪ Comprehensive security appendix in supplier contracts

Technology based security solutions have to complement other domains to achieve the desired security level.

Figure 33: Summary of IoT lecture.

14 Supply Chain Security

Not covered in the lecture.

15 Domain Name System Security

DNS Main function is to provide a mapping of names to resources of several types (e.g. resolving a domain name to an IP address) in a hierarchical and decentralized fashion. It is a globally distributed, loosely coupled, scalable, reliable and dynamic database. Data is maintained locally and retrieved globally.

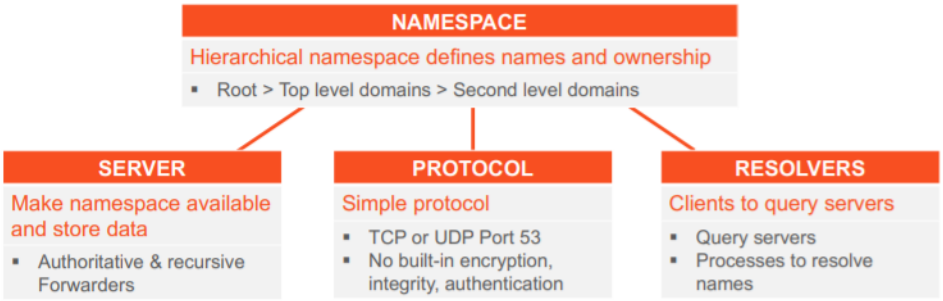

Figure 34: DNS key components.

DNS Security is Critical Manipulating the DNS mapping allows an attacker to redirect connections, perform MITM / impersonation attacks and to launch DoS attacks. For an attacker, a DNS is a freely available distributed storage system or can be used to setup services that are hard to hunt / shut down (e.g. botnets, fast- and domain flux).

DNS Namespace A distributed global lookup mechanism for translating names into other objects organized in a hierarchy (see Figure 35). Authoritative name servers are responsible for mapping domain names to actual Internet resources for each domain (configured by an admin). For each domain, one can define DNS zones that contain a number of distinct sub-level domains (= leaf-nodes).

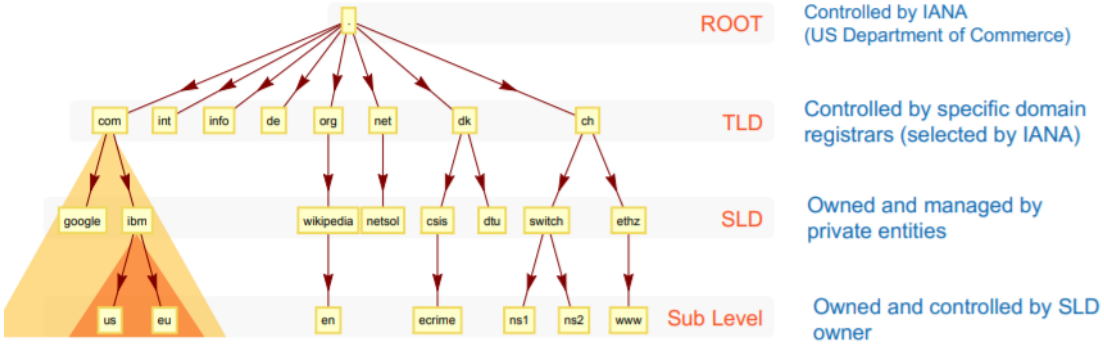

Figure 35: Hierarchical DNS namespace.

DNS Resolution When a client makes a DNS request for a specific domain, it can go through many levels of delegations as depicted in Figure 36.

Resources[6] can be cached at many points during the hierarchical query process. Cache expiration is controlled by the TTL attribute of an entry.

[6] If a domain request couldn't be resolved in the past, it is cached with an *NXDOMAIN* entry, meaning nonexistent domain.

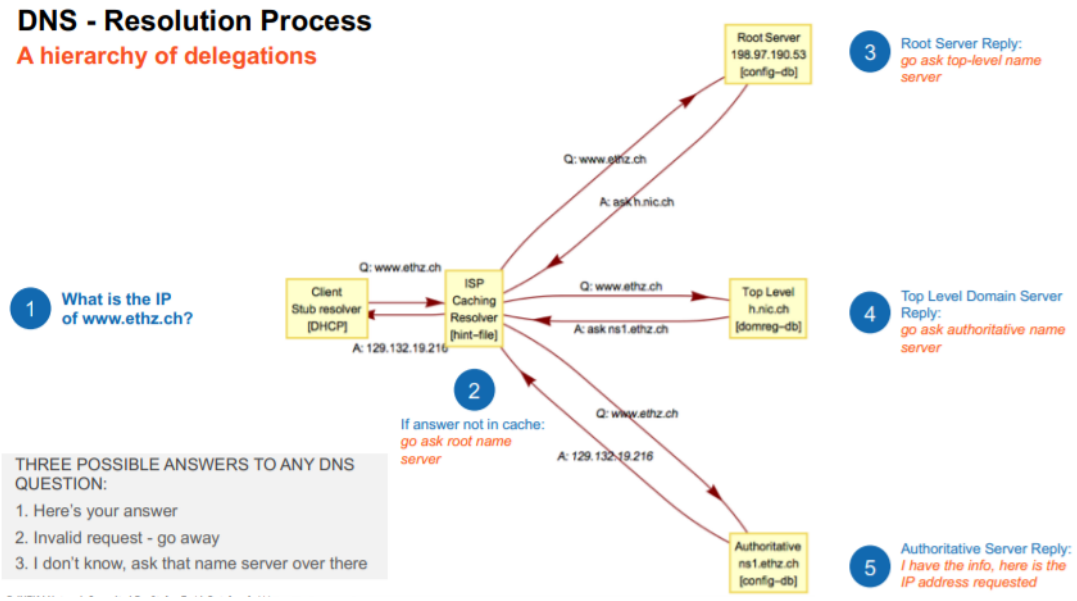

Figure 36: DNS resolution process.

Requests can be resolved recursively, non-recursively or iteratively (or a combination).

Recursive: DNS resolver queries single DNS server which may in turn query other DNS servers. The resolver answers a query completely by querying others as needed.

Non-recursive: DNS resolver queries a DNS server that is either authoritative or provides a partial result.

Iterative: DNS resolver queries a chain of one or more DNS servers. Each server refers client to the next server on the chain until the current one can fully respond.

DNS Key Protocol Features To protect against forged responses, the first two bytes in a message form a transaction ID (*txid*) that must be the same in query and response and introduces 16 bits of entropy (2^{16}) - it is set randomly by the client. DNS messages can either use TCP or UDP (port 53).

Security: If the *txid* does not match, client drops response. Furthermore, client can choose a random source port (16 bits of entropy). There is no confidentiality, integrity verification nor authenticity.

DNS Resource Record (RR) Types RRs define data types in DNS. Each record has a type (name and number), a TTL, a class and type-specific data. See Figure 37 for an overview of the most common ones.

Attack Patterns Objective: insert tampered information into a DNS server or resolution process. See Figure 38 for possible attacks on the DNS resolution process.

Common DNS Attacks

- **Local Host / Network:** Manipulate DNS entries and conversation on local host or network to impersonate services.
- **Cache Poisoning:** Inject manipulated information into DNS cache of resolver to impersonate services.

RECORD TYPE	DESCRIPTION	USAGE
A	ADDRESS RECORD	Maps FQDN into an IP address
PTR	POINTER RECORD	Maps an IP address into FQDN
NS	NAME SERVER RECORD	Denotes a name server for a zone
SOA	START OF AUTHORITY RECORD	Specifies many attributes concerning the zone, such as the name of the domain (forward or inverse), administrative contact, the serial number of the zone, refresh interval, retry interval, etc.
CNAME	CANONICAL NAME RECORD (ALIAS)	Defines an alias name and maps it to the absolute (canonical) name
MX	MAIL EXCHANGER RECORD	Used to redirect email for a given domain or host to another host
TXT	TEXT RECORD	free form text of any type, e.g. Sender Policy Framework (SPF), or and DomainKeys Identified E-mail (DKIM)

Figure 37: DNS resource record types.

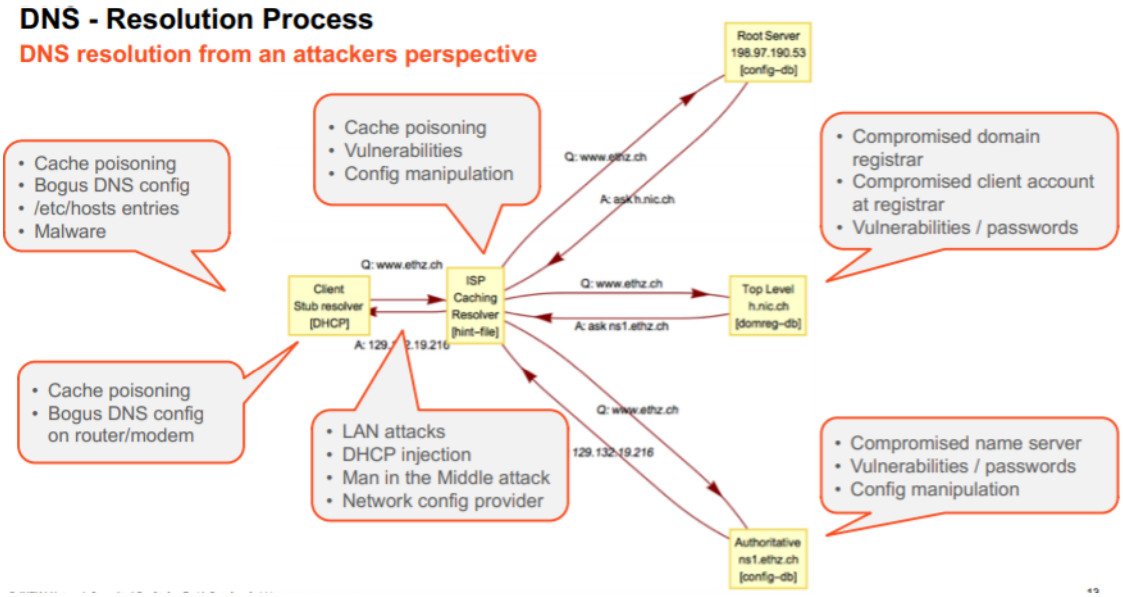

Figure 38: DNS resolution process from an attackers perspective.

- **DNS Tunneling:** Use DNS as a covert communication channel to bypass firewalls to exfiltrate data and hidden communication.
- **DNS Hijacking:** Modify DNS record settings (often at domain registrar) to point to a rogue DNS server / domain to impersonate services.
- **Distributed Reflection:** Abuse a large number of DNS servers to combine reflection and amplification of queries that results in a (D)DoS attack.

DNS Root Server Security Served by 12 root server clusters (A to M) which are authoritative for queries for the top level domains. Every name resolution passes through this.

The root server system (RSS) consists of 1'342 instances and is operated by the 12 independent and diverse (globally and admin) root server operators (RSO). The primary concerns are availability and data integrity of the root zone.

To mitigate a (D)DoS attack on the network bandwidth level, hundreds of root servers are deployed accross different ISPs around the world and they're heavily anycasted.

To mitigate a (D)DoS attack on the memory / CPU level, the RSS is replicated and monitored. DNS enhancements pose a challenge (additional computational overhead).

Cache Poisoning Attacker inserts incorrect resolution information at any level of the resolution process (resolver, forwarder, etc.) and delays the reply of an authoritative name server (expensive lookup or DoS).

It is basically a guessing game for the attacker. An attacker can force a server lookup for a domain (or multiple) and immediately send a number of fake replies that all guess the *txid* and source port number of the request and include a redirect to a malicious IP.

E.g.: Vulnerability patched in 1997 - exploiting the additional section field. By tricking a client to resolve a malicious domain (attacker owns domain), the reply can include unrelated information in an *additional section* field for another valid domain which is then accepted and cached.

E.g.: SADDNS Attack. By forcing a server to send out a query, its source port is public info. Using a port scanner, one can identify the open port (dropped packet = open port, else ICMP port unreachable). Once identified, send large number of spoofed DNS replies to bruteforce *txid* (all while delaying the legit response).

Compromised Configuration Attacking the domain registrar and making it provision wrong information. Second level domains are registered with one of the domain registrars of the top level domain. The DNS info is as secure as the webapp / registration processes / passwords of the registrar and the domain owner.

One can also manipulate the DNS configuration settings on an internal network or local host by attacking routers, attacking DHCP exchange, using malware to change local hosts file, etc.

DNS Security Extensions (DNSSEC) An extension to DNS introduced around 2000 (slow adoption) that provides origin authentication, authenticated denial of existence, integrity but neither availability nor data confidentiality. Zone data is digitally signed using a private key for that zone (parent signs children's public keys). A resolver only needs to know the root public key to authenticate DNS messages. Can have more (list of trust anchors) for zones it trusts implicitly.

To make this work, registrants that are responsible for publishing DNS information must ensure that their data is DNSSEC signed and network operators need to enable DNSSEC validation on their resolvers.

See Figure 39 for some resource record types introduced by DNSSEC.

DNS over HTTPS / TLS (DoH / DoT) Both protocols add confidentiality to DNS. Traditional DNS requests can be used to track a user.

Conclusion

- DNS has suffered a feature creep, picking up increasingly more responsibilities.
- Ensure large enough randomness / entropy for critical fields.
- Consider impact of input validation, rate limiting, max. open / outstanding connections.

RECORD TYPE	DESCRIPTION	USAGE
RRSIG	RESOURCE RECORD SIGNATURE	DNSSEC signature for a record set. Resolvers verify the signature with a public key, stored in a DNSKEY-record.
DNSKEY	PUBLIC KEY RECORD	Contains the public key that a resolver uses to verify DNSSEC signatures in RRSIG-records
DS	DELEGATION SIGNER RECORD	Holds the name of a delegated zone. DS record is placed in the parent zone along with the delegating NS-records. References a DNSKEY-record in the sub-delegated zone.
NSEC	NEXT SECURE RECORD	Contains a link to the next record name in the zone and lists the record types that exist for the record's name. DNS Resolvers use NSEC records to verify the non-existence of a record name and type as part of DNSSEC validation.

CLIENT	SERVER
Indicates DNSSEC support	Include RRSIG signature if DNSSEC is supported

Figure 39: Resource record types added by DNSSEC.

16 Mail Filtering

Guest lecture.

Simple Mail Transfer Protocol (SMTP) Protocol used to send and receive mail messages. For clients, SMTP is used to send emails to a mail server for relaying. To retrieve messages, IMAP (retrieve mail from a mail server) or POP3 are used (or others / proprietary protocols).

Boundary MTA use DNS to look up the MX record for the recipient's domain (after @). MX record contains name of target MTA

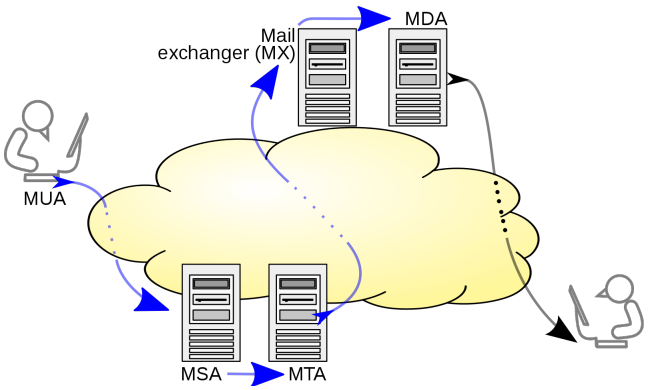

Figure 40: Mail user agent sending an email to a mail submission agent, which in turn sends it to a mail transfer agent.

Mail Filter There are several considerations when designing a mail filter:

- Accept all and filter or already reject some during the SMTP session?
- Retrieve messages during a single SMTP session or multiple parallel ones?
- Filter only inbound mail or also outbound?
- Own DNS server for the filter?
- How to check valid recipient addresses? (local list, remote database, mail server, etc.)

Figure 41: Example of an SMPT message exchange. Note: actual sender and recipient does not match in message content.

- Will users have individual black- and whitelists?
- Will users have individual filtering preferences?

17 Probabilistic Traffic Monitoring

17.1 Overview

Why Monitor Traffic Traffic monitoring can be used to detect anomalies (e.g. volume-based attacks such as a DoS or breaches of quality of service agreements) and to manage the network (usage-based pricing, traffic engineering, etc.).

Performed on a flow-based granularity. Flow = (Src IP, Dst IP, Src Port, Dst Port, Protocol). Also, IPv6 has an explicit flow label.

Difficult since there's so much traffic (and it grows fast) and packets needs to be processed in an extremely efficient manner.

Probabilistic Traffic Monitoring Trade accuracy (low bias) / precision (low variance) for efficiency by estimating traffic statistics based on actual traffic.

17.2 Measuring Flows

General-Purpose Measurements (NetFlow) Standard NetFlow samples every packet, sampled NetFlow samples every k-th packet. It keeps a flow entry for each flow that counts the number of samples packets and bytes which is updated for every sample. To estimate the number of packets and bytes of a flow, the recorded values are multiplied by k.

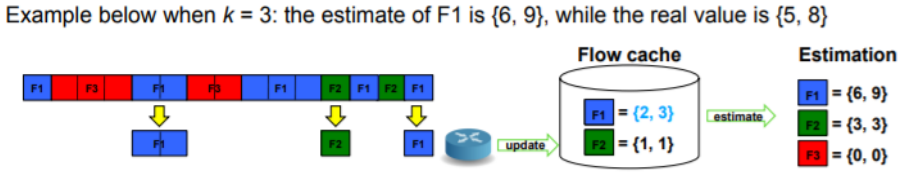

Figure 42: Sampled NetFlow with k = 3.

Simple to implement and reduces the processing time but might be a memory overhead since there is one entry per flow in the worst case. The estimates are imprecise, especially for short-lived flows (most precise for flows sending many small packets). To solve imprecision, only focus on a specific traffic information instead of general purpose (e.g. only large flows as below, etc.)

Large Flows Flows that take up a given threshold of link capacity. E.g. hypergiants that generate 30% of all Internet traffic. Less than 1% of flows account for more than 90% of traffic volume. It is easy to identify them (without keeping per-flow state on routers) because their number is much smaller than overall flows.

Sample and Hold Is a byte-sampling technique that takes packet size into account. Each byte is sampled with a probability p so each packet with size s is sampled with probability p_s where $p_s = 1 - (1-p)^s \approx p \cdot s$ (when p is small). This reduces memory overhead since non-uniform sampling is biased toward large flows.

- For each packet, check if its flow record exists. If yes, hold it (update flow entry). If no, sample it with probability p_s.

- Held / sampled packets are used to update the flow entries in the flow table (estimated number of packets and bytes per flow).

Comparison to NetFlow: Sample-and-Hold uses byte sampling instead of packet sampling. It does not over-count (but both might underestimate). Error rate of S-H is $O(M^{-1})$ and of NF $O(M^{-\frac{1}{2}})$ where M is given memory overhead. S-H inspects all packet headers.

Single-Stage Filter Keep an array of n counters. A router hashes the flow ID of an incoming packet with output i in $\{1, ..., n\}$ and increases i-th counter. Flow is considered large if its counter value surpasses a threshold. To reduce false positives, use multiple filters in parallel.

Multistage Filter The above but with multiple filters in parallel, with each stage using a different independent hash function. Flow is large if all associated counter values surpass a threshold.

Uses fixed memory resources, no false negatives and low false positives (FP rate decreases exponentially in the number of stages).

Finding Frequent Items Find items (= packets) with a frequency that surpass chosen threshold (= large flow). See below for different kinds of algorithms.

Majority Algorithm Keep one item *kept item* and one counter.

First pass For each new item: if counter equals to 0, *kept item* is new item, increase counter, else if new item is same as *kept item*, increase counter, else decrease counter.

Second pass Test the last *kept item* to see if it is actually the one with ¿50% frequency (majority).

MG-Algorithm Generalization of Majority Algorithm. Find all items that appear in a stream of m items more than k times with no false negatives and with limited space $n = \frac{m}{k} - 1$ counters. Needs a second pass to eliminate false positives (e.g. c in example).

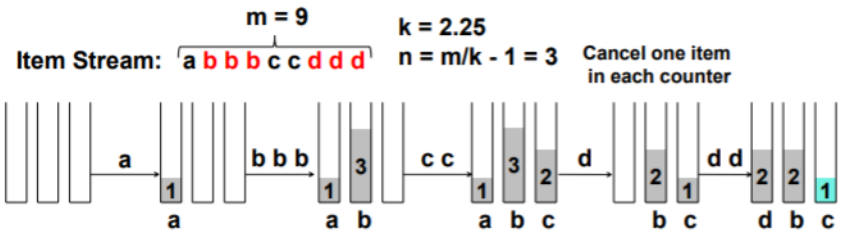

Figure 43: MG-algorithm.

EARDet Algorithm Modifies the MG-Algorithm by switching number of packets with packet size and therefore increasing counter by size instead of number. Idle periods are replaced by virtual flows of a specific packet size (low-bandwidth threshold, e.g. idle period of 6 divided into two times 3 if THL is 3).

No false negatives for large flows and no false positives for small flows (if a certain threshold is surpassed by one of the items in the counter, it is frequent) - there is an ambiguity region between low-bandwidth and high-bandwidth threshold. Deterministic (keeps performance regardless of input traffic / attack pattern). Relatively small storage cost but many counters are needed. Con: per-packet counter is an expensive operation.

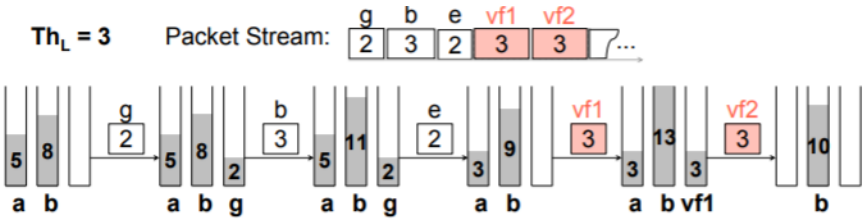

Figure 44: EARDet algorithm.

17.3 Finding Duplicates

Problem Identify if an element is a duplicate without storing all previous elements. A Bloom Filter provides a probabilistic data structure for set membership testing.

Bloom Filter Start with bit vector V that has m bits and an initial value of 0. For each inserted element e, compute k hash functions $h_i = H_i(e)$ where $0 < h_i \leq m$ and set all bits $V[h_i] = 1$. To test if an element e' has already been seen, recompute all k hash functions and check all $V[h_i]$, if all bits are 1, the element has been seen before.

No false negatives and constant time insertion / test. The more elements are inserted, the higher the probability of a false positive (reset filter, no more FN guarantee). Constant bits per element given FP rate (but still O(n) memory overhead).

- m: # of bits in BF
- k: # of hash functions
- n: # of inserted elements
- P(FP) = $\left(1 - \left[1 - \frac{1}{m}\right]^{kn}\right)^k \approx \left(1 - e^{-kn/m}\right)^k$.
- k to minimize P(FP): $k = \frac{m}{n} \ln 2 \approx 0.7\frac{m}{n}$,
- For that choice of k, resulting P(FP) = p = $2^{-k} \approx 0.6185^{m/n}$.
- Given optimal k, choice of optimal $m = -\frac{n \ln p}{(\ln 2)^2}$. → O($n$) space

Figure 45: Bloom Filter parameters.

17.4 Estimate Number of Flows

Probabilistic Counting See Figure 46. Problems: minimum can have very large variance and therefore this is not robust. An attacker controlling only one input can bias the estimation.

Proposal by Bar-Yossef et al.: keep track of the k smallest hash values, expectation value of k-th smallest value is $\frac{k}{(n+1)}$ which has smaller variance. Estimate with k-th smallest value v_k seen so far: $n \approx \frac{k}{v_k} - 1$.

Additionally, use sampling to reduce overhead induced by hashing.

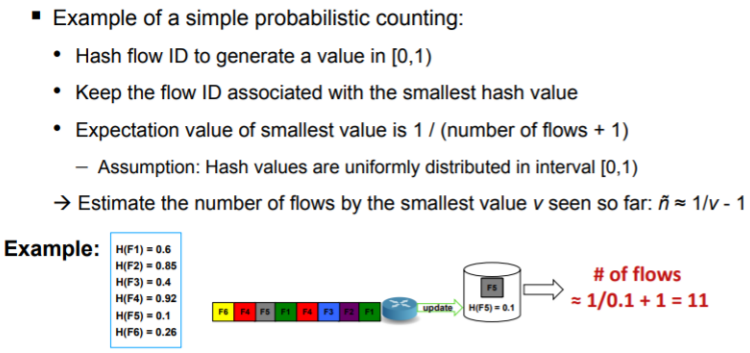

Figure 46: Probabilistic counting.

18 Top N Ways to Get Domain Admin

Guest lecture.

A field report from attack simulations across different companies.

Attacks exploit the human factor almost all of the time (e.g. phishing). You don't even need technical vulnerabilities. Just look for old and outdated stuff.

Attack Simulation The entire network is in scope (company and employees). We don't just test single groups or parts of the company network. Brain goes brrrrrrr....