Network Security: Security, Threats

Tuomas Aura, Microsoft Research, UK

Outline

1. Information security, network security
2. Basic network threats: sniffing and spoofing
3. Cryptography for protocol engineers

Information security, network security

What is security

- When talking about security, we are concerned about bad events caused with malicious intent
  - Security vs. reliability
- Terminology:
  - Threat = bad event that might happen
  - Attack = someone intentionally causes the bad thing to happen
  - Exploit = implementation of an attack
  - Vulnerability = weakness in an information system that enables an attack
  - Risk = probability of an attack × damage in dollars

Areas of IT security

[Gollmann]

- Network security — security of communication
  - Focus: data on the wire
  - E.g. encryption to prevent sniffing
- Computer security — security of end hosts and client/server systems
  - Focus: access control in operating systems
  - E.g. access control lists for file-systems
- Application security — security of services to end users and businesses
  - Focus: application-specific trust relations
  - E.g. secure and legally binding bank transactions

Viewpoints to security

- Cryptography (mathematics)
- Computer security (systems research)
- Network security (computer networking)
- Software security (software engineering)
- Formal methods for security
- Hardware security
- Human aspects of security (usability, sociology)
- Security management (information-systems management, enterprise security)
- Economics of security
- Laws and regulation

Security is a continuous process

- Continuous race between attackers and defenders
  - Attackers are creative
- No security mechanisms will stop all attacks; attackers just move to new paths and targets
  - Some types of attacks can be eliminated but others will take their place
  - Compare with crime statistics: Do locks or prison reduce crime in the long term?
- Security mechanisms will fail and new threats will arise
  ➔ Contingency planning: how to recover from a breach
- Network security is more straightforward than application security, but difficult enough

7

Cost vs. benefit

- Rational attackers compare the cost of an attack with the gains from it
  - Attackers look for the weakest link; thus, little is gained by strengthening the already strong bits
- Rational defenders compare the risk of an attack with the cost of implementing defences
  - Lampson: "Perfect security is the enemy of good security"
- But human behavior is not always rational:
  - Attackers follow each other and flock all to the same path
  - Defenders buy a peace of mind; avoid personal liability by doing what everyone else does

→ Many events are explained better by group behavior than rational choice

8

Proactive vs. reactive security

- Technical prevention: design systems to prevent, discourage and mitigate attacks
  - If attack cannot be prevented, increase its cost and control damage
- Detection and reaction: detect attacks and take measures to stop them, or to punish the guilty
- In open networks, attacks happen all the time
  - We can detect port scans, spam, phishing etc., yet can do little to stop it or to punish attackers

  ➔ Technical prevention and mitigation must be the primary defence
- However, detection is needed to monitor the effectiveness of the technical prevention

9

Network Security Goals

- Confidentiality — no sniffing
- Authentication and integrity — no spoofing of data or signaling, no man-in-the-middle attacks
- Access control — no unauthorized use of network resources
- Availability — no denial of service by preventing communication
- Privacy — no traffic analysis or location tracking

10

Authentication and integrity

- Peer-entity authentication = verify that presence and identity of a person, device or service at the time; e.g. car key
- Data origin authentication = verify the source of data
- Data integrity = verify that the data was received in the original form, without malicious modifications
- In practice, data origin authentication and integrity check always go together
- Authentication (usually) requires an entity name or identifier

11

Network security mechanisms

- Strong cryptography:
  - Encryption → confidentiality
  - Cryptographic authentication
    → entity authentication and access control
    → data authenticity and integrity
- Non-cryptographic mechanisms:
  - Over-provisioning
  - Perimeter defense (firewalls)
  - Stopping attacks at source
  - Proxies and pseudonyms
  - Intrusion detection
- Non-technical solutions:
  security is also a social, legal and business problem
  (but that is not the topic of this course)

12

Who is the attacker?

- We partition the world into good and bad entities
  - Honest parties vs. attackers
  - Good ones follow specification, bad ones do not
  - Different partitions lead to different perspectives on the security of the same system
- Typical attackers:
  - Curious or dishonest individuals — for personal gain
  - Hackers, crackers, script kiddies — for challenge and reputation
  - Companies — for business intelligence and marketing
  - Security agencies — NSA, FAPSI, GCHQ, DGSE, etc.
  - Military SIGINT — strategic and tactical intelligence, cyberwar
  - Organized criminals — for money
- Often, not all types of attackers matter
  - E.g. would you care if NSA/university/mom read your email?

13

Protocol engineering

- Network is a distributed system with many participants
- Computer networking is about protocols
  - Protocol = distributed algorithm
  - Algorithm = stepwise instructions to achieve something
- Security is just one requirement for network protocols
  - Cost, complexity, performance, deployability, time to market etc. may override perfect security
- Like the design of cryptographic algorithms, security engineering requires experienced experts and peer scrutiny
  - Reuse well-understood solutions; avoid designing your own
- The most difficult part is understanding the problem
  - Must understand both security and the application domain
  - Potential solutions often become obvious

14

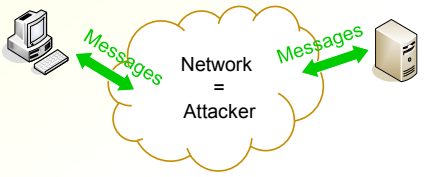
Security vs. cryptography

- In the following lectures, we will use cryptography as the main building block for security protocols
- However:
  "Whoever thinks his problem can be solved using cryptography, doesn't understand the problem and doesn't understand cryptography." — attributed to Roger Needham and Butler Lampson

15

Basic network threats: sniffing and spoofing

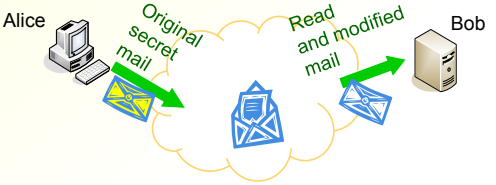
Traditional network-security threat model



- End are nodes trusted, the network is unreliable
- End nodes send messages to the network and receive messages from it
- Network will deliver some messages but it can read, delete, modify and replay them
- Metaphors: unreliable postman, notice board, rubbish basket

17

Example: email



- Alice sends an email that is addressed to Bob
- The attacker may read, delete and edit the email. It may copy the email, or cut and paste pieces from one email to another. It may write a new email
- Secrets and message integrity need protection

18

Basic network security threats

- Traditional major threats:
  - Sniffing = attacker listens to network traffic
  - Spoofing = attacker sends unauthentic messages
  - Data modification (man in the middle) = attacker intercepts and modifies data
- Corresponding security requirements:
  - Data confidentiality
  - Data-origin authentication and data integrity

19

Sniffing

- Sniffing = eavesdropping = spying = unauthorized listening = monitoring
- Sniffers:
  - Packets are often broadcast on a local link → all local nodes can listen
  - Sniffers listen to packets on the network and pick out interesting details, e.g. passwords
  - Hackers install sniffer software on compromised hosts; tools are available for download
  - Wireless Ethernet most vulnerable; sniffing on switched LANs and core networks is more difficult but possible
- Network admins and spies can monitor packets on routers, firewalls and proxies
  - Router security may become a serious issues

20

Spoofing

- Spoofing = sending unauthentic messages = using false sender address or identifier
- In the Internet, it is easy to send messages that appear to come from someone else
  - A modified version of the application or protocol stack is easy to write
- Examples:
  - Email spoofing: false From field
  - IP spoofing: false source IP address
  - DNS spoofing: false DNS responses
  - Mobile-IP BU spoofing: false location information

21

Example: email spoofing

- SMTP does nothing to authenticate the sender

```
> telnet smtp.ntlworld.com 25
220 mta06-svc.ntlworld.com ESMTP server (InterMail vM.4.01.03.27
201-229-121-127-20010626) ready Tue, 11 Mar 2003 20:53:02 +0000
mail from:President <president@whitehouse.gov>
250 Sender <president@whitehouse.gov> Ok
rcpt to:Tuomas Aura<tuomaura@microsoft.com>
250 Recipient <tuomaura@microsoft.com> Ok
data
354 Ok Send data ending with <CRLF>.<CRLF>
To: Tuomas Aura
Subject: Hello!

May the force be with you.
.
250 Message received: 20030311214331.FQKK2273.mta03-
svc.ntlworld.com@[80.4.4.33]
quit
221 mta06-svc.ntlworld.com ESMTP server closing connection
Connection to host lost.
>
```

22

Example: IP spoofing

- Attacker sends IP packets with false source address.
  - Anyone can write software to do this with raw sockets
- The destination node usually believes what it sees in the source address field
- Attacker may be anywhere on the Internet
- Spoofing a connection is more difficult:
  - Attacker must sniff replies from B in order to continue the conversation
  - → Attacker must be on the route between A and B, or control a router on that path

23

TCP sequence numbers and IP spoofing

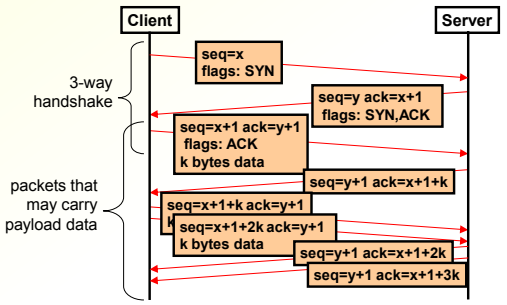
- TCP sequence numbers are initialized to random values during the connection handshake
- Acknowledgment number in the third packet must be sequence number of the second packet + 1
- Sequence numbers are incremented for each byte sent. Packets must arrive in order
- Receiver rejects packets with incorrect sequence numbers and waits for the correct ones

→ TCP packets are difficult to spoof because the attacker must sniff or guess the sequence number

- Not cryptographically secure – receiver may accept individual spoofed packets if attacker guesses right
- The first packet (SYN) is easy to spoof

24

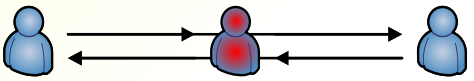
TCP handshake



25

Man in the middle

- In the man-in-the-middle attack, the attacker is between the honest endpoints



- Attacker can intercept and modify data
  → combines sniffing and spoofing
- On the Internet, a MitM attacker must
  - be at the local network of one of the end points
  - be at a link or router on the route between them, or
  - change routing to redirect the packets via its own location

26

Discussion

- What other threats and security requirements are there on open networks?

27

Other network threats

- Other threats:
  - Unauthorized resource use (vs. access control)
  - Integrity of signalling and communications metadata
  - Denial of service (DoS) (vs. availability)
  - Traffic analysis, location tracking
  - Lack of privacy
  - Software security
- Not captured well by the traditional network-security model

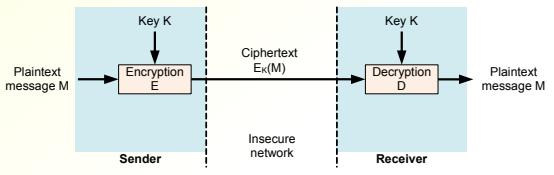
28

Discussion

- What kind of tools and techniques do we have available to protect communication?

29

Encryption



- Message encryption based on symmetric cryptography
  - Endpoints share a secret key K
  - Block ciphers, stream ciphers
- Notations: $E_K(M)$, E(K;M) , $\{M\}_K$, K{M}
- Protects confidentiality, not integrity

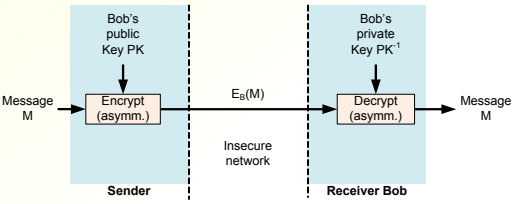
31

Ciphers and modes

- Block ciphers:
  - 3DES in EDE mode: $DES_{K3}(DES^{-1}_{K2}(DES_{K1}(M)))$ — 168-bit keys but only 112-bits of security
  - AES — 128-bit keys
- For messages longer than one block, a block-cipher mode needed, e.g. CBC
- Random initialization vector (IV) makes ciphertexts different even if the message repeats
- Stream ciphers:
  - XOR plaintext and a keyed pseudorandom bit stream
  - RC4: simple and fast software implementation
- Always assume that encryption is malleable
  - Attacker can make controlled modifications to the plaintext
  - Exception: new AES modes with strong authentication

32

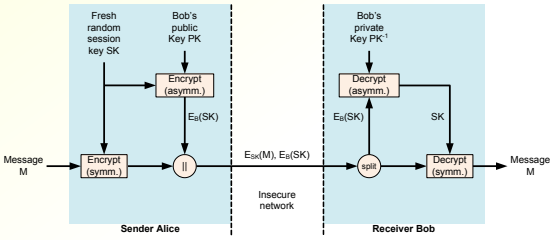
Public-key encryption



- Message encryption based on asymmetric crypto
  - Key pair: public key and private key
- Notation: PK, PK^{-1}, K^+, K^- ; $E_B(M)$, PK{M}, $\{M\}_{PK}$

33

Hybrid encryption



- Symmetric encryption is fast; asymmetric is convenient
- Hybrid encryption = symmetric encryption with random session key + asymmetric encryption of the session key

34

Key distribution

- Main advantage of public-key protocols is easier key distribution
- Shared keys, symmetric crypto:
  - $O(N^2)$ pairwise keys need for N participants → does not scale
  - Keys must be kept secret → hard to distribute
- Public-key protocols, asymmetric crypto:
  - N key pairs needed, one for each participant
  - Keys are public → can be posted on a bulleting board
- Both kinds of keys must be authentic
  - How does Alice know it shares K_{AB} with Bob, not with Trent?
  - How does Alice know PK_B is Bob's private key, not Trent's?

35

Formal security definitions

- Cryptographic security definitions for asymmetric encryption
- Semantic security
  - Computational security against a ciphertext-only attack
- Ciphertext indistinguishability
  - IND-CPA — attacker submits two plaintexts, receives one of them encrypted, and is challenged to guess which it is ⇔ semantic security
  - IND-CCA — indistinguishability under chosen ciphertext attack i.e. attacker has access to a decryption oracle before the challenge
  - IND-CCA2 — indistinguishability under adaptive chosen ciphertext attack i.e. attacker has access to a decryption oracle before and after the challenge (except to decrypt the challenge)
- Non-malleability
  - Attacker cannot modify ciphertext to produce a related plaintext
  - NM-CPA ⇒ IND-CPA; NM-CCA2 ⇔ IND-CCA2

36

Cryptographic hash functions

- Message digest, fingerprint
- Hash function: arbitrary-length input, fixed-length output
- One-way = pre-image resistant: given only output, impossible to guess input
- Second-pre-image resistant: given one input, impossible to find a second input that produces the same output
- Collision-resistant: impossible to find two inputs that produce the same output
- Examples: MD5, SHA-1, SHA-256
- Notation: h(M), hash(M)

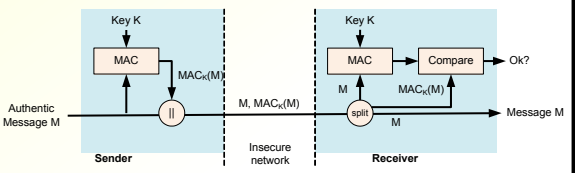
37

Hash collisions

- 128–160–256-bit hash values to prevent birthday attack
- Recent research has found collisions in standard hash functions (MD5, SHA-1)
- Currently, any protocol that depends on collision-resistance needs a contingency plan in case collisions are found
- Security proofs for many cryptographic protocols and signature schemes depend on collision resistance because it is part of the standard definition for hash functions
- However, most network-security applications of hash functions do not really need collision resistance, only second-pre-image resistance

38

Message authentication code (MAC)



- Message authentication and integrity protection based on symmetric cryptography
  - Endpoints share a secret key K
  - MAC appended to the original message M
  - Common implementations: HMAC-SHA1, HMAC-MD5
- Notations: $MAC_K(M)$, MAC(K;M), $HMAC_K(M)$

39

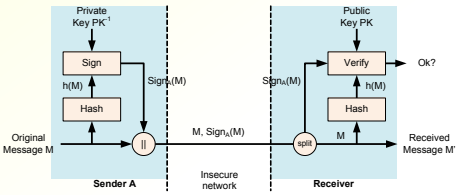
HMAC

- HMAC is commonly used in standards:
  - Way of deriving MAC from any cryptographic hash function h

  $HMAC_K(M) = h((K \oplus opad) \| h((K \oplus ipad) \| M))$
  - Hash function h is instantiated with SHA-1, MD5 etc. to produce HMAC-SHA-1, HMAC-MD5,...
  - $\oplus$ is XOR; $\|$ is concatenation of byte strings
  - ipad and opad are fixed bit patterns
  - Details: [RFC 2104][Bellare, Canetti, Krawczyk Crypto'96] *
- HMAC is theoretically stronger than simpler constructions: $h(M \| K)$, $h(K \| M \| K)$
- HMAC is efficient for long messages; optimized for pre-computation
- Discussion: does h need to be collision resistant?

40

Digital signature (1)



- Message authentication and integrity protection with public-key crypto
  - Verifier has a public key PK ; signer has the private key PK^{-1}
  - Key pair is often associated with a user: PK_A , PK^{-1}_A
  - Messages are first hashed and then signed
  - Examples: DSS, RSA + SHA-256

41

Digital signature (2)

- Examples: DSA, RSA [PKCS#1]
- Public/private key notations:

  $PK = PK_A = K_A = K^+ = K^+_A = e_A$; $PK^{-1} = PK^{-1}_A = K^- = K^-_A = d_A$
- Signature notations:

  $S_A(M) = Sign_A(M) = S(PK^{-1}; M) = PK^-_A(M) = \{M\}_{PK^{-1}}$
- Digital signature with appendix:
  - Signature does not contain the original message M
  - Signatures can be stored separately of M
  - Can append multiple signatures to the same M
  - However, signatures may reveal something of M
  - Historically, there were also signatures with message recovery, in which the signature contains the signed message (e.g., RSA without hashing)
- Discussion: does h need to be collision resistant?

42

Message size

- Authentication increases the message size:
  - MAC takes 16–32 bytes
  - 1024-bit RSA signature is 128 bytes
- Encryption increases the message size:
  - IV for block cipher takes 8–16 bytes
  - 1024-bit RSA encryption of the session key is 128 bytes
- Overhead of headers, type tags etc.
- Size increase ok for most application-level protocols
  - Signing individual IP packets (1500 bytes) is expensive
  - Signing data on wireless connections may be expensive

43

The first broken protocol

- What is wrong with this protocol:
  A → B: M,S_A(M)
  E.g., S_A("Attack now!")

44

Timestamps (1)

- Replay problem:
  A → B: M, S_A(M) // S_A("Attack now!")
- Checking freshness with A's timestamp:
  A → B: T_A, M, S_A(T_A, M)
- Fresh = recently sent, or not received before
- Valid = accepted by recipient
- Timestamp implementations:
  - Sender's real-time clock value (validity ends after fixed period)
  - Validity period start and end (or start and length)
  - Validity period end time
- Notation: T_A

45

Timestamps (2)

- What problems remain?
  A → B: T_A,M, S_A(T_A, M)
  E.g. S_A("Attack now!")
- Timestamps require clocks at the signer and receiver, and secure clock synchronization
- Secure fine-grained synchronization is hard to achieve, loose synchronization (accuracy from minutes to days) is easier
- Also, fast replays possible: S_A("Transfer £10.")

46

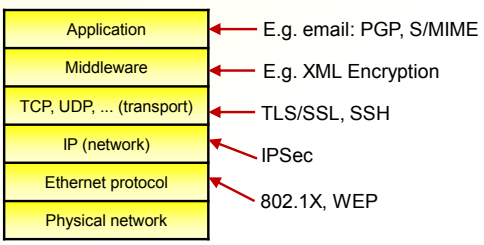
Using cryptography

- Hashing and signing are generally more useful than encryption
  - When old protocol specs say "encryption", they sometimes mean a MAC, too
- Signing is not encryption with private key!
- Algorithm suites and negotiation
  - How many alternatives are needed?
- Cryptography vs. protocol design
  - Security protocol designers can treat crypto algorithms as black boxes BUT…
  - Algorithm properties are often misunderstood
  - Creative use of crypto algorithms is dangerous
  - Must follow crypto research to know if the algorithms are still secure

  → Stick to the very basic algorithms and security properties

47

Protocol Stack and Security



- Security solutions exist for every protocol layer
- Layers have different security and performance trade-offs, trust relations and endpoint identifiers

48

Which layer security?

- Security mechanisms exist for all protocol layers
  - Which layer is right for encryption and authentication?
  - Which layer PDUs should a firewall filter?
- Reasons to implement cryptographic security in lower layers:
  - Security provided by physical, link or network layer is a service to all higher layers
  - Lower-layer security protects *all* higher-layer data
  - Security in lower layers is transparent to higher layers. No changes to applications needed
  - Lower-layer security protects the lower layer, too
- Reason to implement security in higher layers:
  - Security implemented in the application or middleware will fit exactly to the application requirements
  - Lower-layer identifiers may not be meaningful to higher layers
- Actually, we may need independent security in multiple network-stack layers

49

Arguments for end-to-end Security

- Confidentiality and authenticity are usually user or application requirements
- Link-layer security assumes all routers are trusted
  - The end nodes have to be trusted anyway. It is unnecessary to trust intermediaries
- Link-layer security is different for each link type
- End-to-end security is sufficient to provide confidentiality and authentication for applications. Building a secure network is unnecessary
  - Network only needs to protect itself, not the application data

50

Exercises

- Design a more spoofing-resistant acknowledgement scheme to replace TCP sequence numbers. Hint: use random numbers to ensure that acknowledgements can only be sent by someone who has really seen the packets
- Which applications of hash functions require collision resistance?

51