Computer Security

16r. Pre-exam review

Paul Krzyzanowski • David Domingo • Ananya Jana

Rutgers University

Spring 2019

This covers some highlights of the past four lectures – *not* all the material

If any of this is really unclear to you, it's an indication that you should spend some time studying the material

Authentication

Authentication

- Factors
  1. Something you **have** (key, card, phone, USB dongle)
  2. Something you **know** (password, PIN)
  3. Something you **are** (biometrics)
- Multi-factor authentication
  - Using more than one of these factors
  - E.g., Password + card

Protocols: Reusable Passwords

Password Authentication Protocol (PAP)

- Classic { *username, password* } validation
- **Hashed** passwords
  - Storing hash(password) ensures that attackers won't see passwords if they get hold of the password file
- **Salted** hashes
  - Adding random text (salt) to a password before hashing it guards against dictionary attacks

Protocols: One-Time Passwords

1. Sequence-based: password = *f*(previous password)
   - Example: S/key authentication
2. Time-based: password = *f*(time, secret)
   - Example: Time-based One-Time Passwords (TOTP)
3. Challenge-based: *f*(challenge, secret)
   - Example: Challenge-Handshake Authentication Protocol (CHAP)

U2F: Universal 2nd Factor Authenticator

- Hardware authenticator (usually Bluetooth or USB)
  - Stores public/private keys for each service
- Uses challenge-based authentication
- **Registration with a service** (usually a web site) – initial access
  - Server sends a challenge (*N*)
  - Device generates public/private key pair for the service
  - Sends *{ device_id, public key, N }* signed with its private key
- **Authentication**
  - Server sends a challenge (*N*)
  - Device sends back *{ device_id, N }* signed with its private key
  - Server can validate using the public key associated with that *device_id*

Code Signing

Challenge: distribute software and ensure that it is not modified during distribution or on the computer

Solution
- Use digital signatures, just like for network messages
- **Publisher**: Hash the software → encrypt the hash with your private key
  - X.509 certificate attached to the application
- **OS**: Hash the software → validate the hash using the publisher's public key

- Per-page signatures: sign page-size blocks of software
  - OS can verify a page as it is loaded instead of scanning the entire file ahead of time

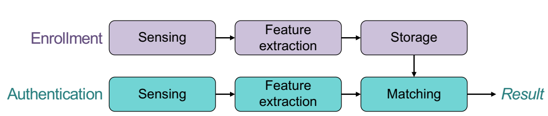
Biometrics

Biometric Authentication

- Identify a person based on physical or behavioral characteristics
  - Not ownership of keys or knowledge of passwords
- Unlike other forms of authentication
  - Biometrics relies on statistical pattern recognition
  - Comparing sampled biometric data with stored biometric data will almost never yield an exact match
  - Software will decide whether the matches are "good enough"
    - False Accept Rate (FAR): false match
      - Non-matching pair of biometric data is *accepted* as a match
    - False Reject Rate (FRR): false non-match
      - Matching pair of biometric data is *rejected* as a match

Authentication Process

1. Sensing
   - Capture the biometric data
2. Feature extraction
   - Extract the interesting (unique) parts of the data
3. Pattern matching
   - Compare the extracted data with stored samples
4. Decision
   - Decide whether the sensed data is close enough to the stored sample



CAPTCHA

- Not biometrics – a technique for software to detect if it's dealing with a human being or a bot
  - Present distorted text (or images) that is difficult for a computer to process but relatively easy for humans
- Problem: OCR & computer vision has improved to the point where computers can match human skill
- NoCAPTCHA RECAPTCHA
  - No puzzles!
  - Perform "risk analysis"
    - Check IP address of known bots
    - Check Google cookies for legitimate users
    - Track mouse movements for randomness

Network security

Data link layer

- MAC Attacks – **CAM overflow**
  Sniff all data on the local area network
  – If you send spoofed random source addresses, you will overflow the Ethernet switch's table stored in content-addressable memory (CAM)
  – The switch will then broadcast all traffic onto all ports – *enables sniffing traffic*
- **VLAN Hopping**
  Sniff all data from connected virtual local area networks
  – A computer can spoof itself to appear as an ethernet switch with a trunk connection to another switch
  – It will receive traffic for all VLANs (Virtual Local Area Networks) and can see all of it rather than just the traffic for one VLAN

Data link layer

- **ARP cache poisoning**
  Redirect IP packets by changing the IP address → MAC address mapping
  – Address Resolution Protocol (ARP): computer broadcasts a query asking if anyone knows the MAC address corresponding to a given IP address
  – If a malicious host responds with its MAC address, it will receive traffic for that IP address
- **DHCP server spoofing**
  Configure new devices on the LAN with your choice of DNS address, router address, etc.
  – Assigns IP address, subnet mask, router address, DNS server address
  – A malicious host can act as a DHCP server and provide bad data for routers or DNS servers to redirect traffic

Network (IP) & transport (TCP/UDP) layers

No source address authentication – anyone can fake a source address

- **UDP data**– trivial to forge since there is no sequencing
- **TCP data** – harder: need to match sequence numbers
- TCP connection setup
  – **Random starting sequence numbers** make it hard to guess sequence #
  – **SYN flooding attack**: *Denial of Service*
    - Send TCP connection requests (SYN packets) with an unreachable source address
    - Receiver will allocate resources for the connection
    - Eventually will not be able to accept any more connections
  – Defense: **SYN cookies:** *minimize SYN flooding problem*
    - Do not allocate resources until the handshake is complete
    - Server computes the SYN-ACK sequence number by
      – hash(src_addr, dest_addr, src_port, dest_port, SECRET)
      – SECRET is just a random number that the server picked

Routing Protocols & DNS

- **BGP** (Border Gateway Protocol)
  – Used by IP networks (autonomous systems) to share **routing** information
  – Uses a TCP connection between routers
  – Route announcements are not authenticated
  Attacks
  – **Fake route announcements** can cause routers throughout the Internet to redirect data to a different place
- **DNS** (Domain Name System)
  – Responsible for converting domain names to IP addresses
  Attacks
  – Responses can be intercepted & modified, providing the wrong address for a domain name

Blockchain & Bitcoin

The blockchain

- Decentralized list of transactions (***ledger***)
  - **Block** = set of transactions (in Bitcoin, ~10-minute window)
  - **Blockchain**: blocks connected via **hash pointers** into a list of blocks
  - Entire blockchain is replicated on all participating servers
  - **Merkle tree**: binary tree of hash pointers within a block to make it easy to find the desired transaction
- User ID (**address**) = your public key
  - Only you have the private key (which is stored in your **wallet**)
- Guarding against forgery
  - Each transaction signed by the owner

Avoiding double spending

- New transactions are sent to all participants
- Each transaction identifies **inputs** (past transactions where the money comes from)
  - No two transactions cannot use the same inputs
  - This ensures there is no **double spending**
- Each participant checks the blockchain to make sure the transaction is valid
- Valid transactions are added to the block

Proof of Work

When a block is ready to be added to the chain…

Secure the block with a **Proof of Work**

- Field in the block that is modified so that the *hash(block)* has specific properties (first *n* bits are 0).
- This takes a huge amount of computation – trying different bit patterns

Finding the Proof of Work is called **mining**

Proof of Work

- The **first server** that computes the Proof of Work advertises it to other systems
  - Each receiver **validates**: this is efficient – just compute the hash
  - Server that finds this gets rewarded with bitcoins
- When a **majority** of systems approves the Proof of Work
  - The block becomes part of the blockchain (connected via a hash pointer to the previous block)

Changing the Past

The Proof of Work makes it difficult for a server to change old transactions

- You would need to recompute the Proof of Work for all blocks back to the one you need to modify
- This means creating an alternate blockchain

- If there are competing blockchains
  - The **longest chain** is considered the legitimate one
- **51% attack**
  - To alter past transactions & create a longer chain, you need to own over 50% of the computation power

Confirming transactions

When do we feel safe about a transaction?

- A transaction is ***confirmed*** after *N* number of additional blocks are added to the blockchain

- A confirmation value of ***N*** mean that an attacker will need to recompute ***N+1*** Proof of Work values to modify the blockchain
  - Computationally not feasible
  - Large values of *N* are recommended for high-value transactions (typically N=6)

Firewalls & VPNs

Virtual Private Networks

- Key principle: **Tunneling**
- **IPsec** – popular set of VPN protocols
  - **Authentication Header (AH) protocol**
    - Guarantees integrity & authenticity of IP packets – does not encrypt
  - **Encapsulating Security Payload (ESP)**
    - Adds encryption of the entire payload (encapsulated packet)
- IPsec uses
  - **Authentication**: Digital certificates or pre-shared keys
  - **Key exchange**: Diffie-Hellman
  - **Confidentiality**: Symmetric cryptography
  - **Integrity**: HMAC (hash-based MACs)

Transport Layer Security (TLS)

Goal: provide an authenticated, encrypted, and tamper-proof connection at the transport layer between two hosts that software can use in a manner similar to TCP sockets

- **Authentication**
  - Use public key cryptography & X.509 certificates for authentication
- **Key exchange**
  - Diffie-Hellman keys generated per session (or pre-shared keys)
- **Confidentiality**
  - Use symmetric cryptography to encrypt data
- **Integrity**
  - Include an HMAC with transmitted data to ensure message integrity
- Support different key exchange, encryption, integrity, & authentication protocols – negotiate what to use at the start of a session

Firewalls

Firewall (screening router)	1st generation packet filter that filters packets between networks. Blocks/accepts traffic based on IP addresses, ports, protocols
Stateful inspection firewall	Like a screening router but also takes into account TCP connection state and information from previous connections (e.g., related ports for TCP)
Application proxy	Gateway between two networks for a specific application. Prevents direct connections to the application from outside the network. Responsible for validating the protocol.
IDS/IPS	Can usually do what a stateful inspection firewall does + examine application-layer data for protocol attacks or malicious content
Host-based firewall	Typically screening router with per-application awareness. Sometimes includes anti-virus software for application-layer signature checking
Host-based IPS	Typically allows real-time blocking of remote hosts performing suspicious operations (port scanning, ssh logins)

Web Security

Same-origin policy

- Basic security model for the web
  - A browser permits scripts in one page to access data in a second page ***only if*** both pages have the same origin
  - **Origin** = { URI scheme, hostname, port number }
- Each frame gets the origin of its URL
  - JavaScript code executes with the authority of its frame's origin
  - If cnn.com loads JavaScript from jQuery.com, the script runs with the authority of cnn.com
  - Passive content (CSS files, images) has *no* authority
    - It doesn't (and shouldn't) contain executable code
- Cross-Origin Resource Sharing (CORS)
  - A way for the server to tell a browser to treat multiple origins as the same

Some browser attacks

Cross-Site Request Forgery (CSRF)	• Browsers send all relevant cookies to the server with each request – these often contain login information • Some services place commands on the URL • *Attack: An attacker may get you to take some action on a site where you are already authenticated by getting you to click on a link*
Cross-Site Scripting (XSS)	• Code injection attack: malicious scripts are added as part of user input and later presented back to a user. • *Reflected XSS: attacker creates a malicious link. User clicks on it and the response goes back to the user's browser with the malicious script in it.* • *Persistent XSS: attacker adds malicious JavaScript where it will be stored on a server and presented to other users (e.g., forum comments)*
Clickjacking	• *Attacker overlays an image to trick a user to clicking a button or link – the user clicks on something different than they think they're clicking on*

Mobile Device Security

Android Security

- App isolation
  - Linux user IDs are used as app IDs: each app has its own Linux UID
  - Java apps run in a Dalvik virtual machine
- Mandatory code signing
  - Can be self-signed or signed by a third party – Android does not validate CAs
- App communication
  - Apps communicate with ***intents***: messages that contain an action & data sent to some other component
    - This is the way apps request services from system services or other apps
- Permissions
  - Apps must request permission to access resources at install time
  - OS maintains a whitelist of what an app is allowed to access
- File system encryption

iOS Security

- App isolation
  - App sandbox restricts access to other app's data & resources
- App communication
  - Inter-app communication only through iOS APIs
- Mandatory code signing
  - Must be signed using an Apple Developer certificate
- App data protection
  - Apps can use built-in hardware encryption
- File encryption
  - Each file is encrypted with a unique key

Hardware protection

- ARM TrustZone: two "worlds"
  - Non-secure world
    - Cannot access secure resources directly
    - Main OS and apps run in the non-secure (non-trusted) world
  - Secure world
    - Cryptographic functions & key storage
- Each world has its own OS & applications
  - Secure key management & key generation
  - Secure boot, digital rights management, secure payment
- Apple Secure Enclave: Apple's customized TrustZone-like solution
  - Dedicated co-processor for the secure world
  - All cryptographic functions are handled in the secure enclave (secure world)

The end

