CS161 Computer Security MT1

1. Principles of Security

Security is Economics

- Least Privilege - give least amount of privilege needed
- Use Fail Safe defaults - deny all access and only allow those that are explicitly permitted
- Separation of Responsibility - split up privilege so no one person or program has complete power. Require more than one party to approve before access is granted
- Ensure complete mediation - when enforcing access control policies, make sure that you check every access to every object
- Know your threat model - be careful with old code. the assumptions originally made might no longer be valid. The threat model may have changed.
- Conservative design -
- Kerchoffs Principle - should be secure even after knowing all details
- Shannon's Maxim - the attacker knows the system - security through obscurity
- Defense in depth - multiple checks
- consider human factors - security systems should be usable
- psychological acceptability - user buys into the model

2. Software Security

Registers

- EIP / RIP: Instruction Pointer
- EBP: base pointer, top of frame
- ESP: stack pointer, bottom of frame

When the next frame is called

→ Return Address: intuitively points to where the function will return to

→ SFP: Stack Frame Pointer, saved directly below return address

→ EBP: bottom of the previous frame is the top of this frame

- EBP stays fixed
- ESP moves around

STACK

args
RIP
SFP
locals arguments

stack ↓

↑ string

Canary

- place a canary right after SFP
- makes it harder for smash stack
- new canary each program start

ASLR

- randomize the address of each chunk of memory each time
- most ASLR do not randomize text location. all address spacer randomized. Stack, Heap, BSS, DATA

Attacks

- RoP
- NoP sled + Ret2Ret
- Ret2ESP
- Ret2ESX
- bruteforce

NXBit (W^X, DEP)

- Non executable stack. can't place shellcode in stack buffer.

Attack

- ROP
- local variable manipulation

TOCTTOU

* file open, read

OFFBYONE

- change SFP last byte to null, so caller's epilogue will go to RIP 1 word after changed SFP. Place shellcode there.

def: invariant: things that are always true and won't change "if this was true before you called me, I promise it'll still be true when done"

def: pre-conditions: things that must be true before a method is called

def: post-conditions: things that must be true after method is complete

3. Cryptography

XOR Basics

1) $x \oplus 0 = x$ (0 identity)
2) $x \oplus x = 0$ (x is its own inverse)
3) $x \oplus y = y \oplus x$ (commutative)
4) $(x \oplus y) \oplus z = x \oplus (y \oplus z)$ (associative)

Symmetric key cryptography

- **IND-CPA (indistinguishability)**
  1. adversary choose m_0, m_1 of same length
  2. Challenger choose random bit b $\{0,1\}$ and encrypt m_b
  3. if adversary guesses b with $P(1/2 + negl)$ then IND-CPA

ONE TIME PAD

- get key k, message m, $C = Enc(k, m)$
- $m = Enc(k, C)$

Block Cypher

def: secure :

E_k = random permutation

Attacker

Referee

- randomly pick one Box, and encrypt using that m, Box
- secure if attacker guesses which box used for encryption

1. Alice & Bob share key K
2. 2 Messages are not encrypted the same] Goal
3. Encrypt long messages

1. ECB Mode (Electronic code book): plaintext M is broken into n-bit blocks $M_1, \cdots, M_t$ and each block is encoded using block cypher : $C_i = E_k(M_i)$

   $C = C_1 | C_2 | \cdots | C_t$

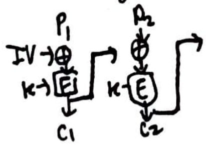
   Flawed: NOT IND-CPA, two same messages encrypt to same

2. CBC Mode (Cipher Block chaining): Popular mode for commercial apps. For each message the sender picks a random n bit string called initial vector (IV).

   Define $C_0 = IV$. The ith ciphertext block is given by

   $C_i = E_k(C_{i-1} \oplus M_i)$

   $C = IV | C_1 | C_2 | C \cdots | C_t$



   Proven to be strong

   Pros : generate random IV

   cons : cant parallelize, cant repeat IV, same m, will enc to same

   if IV is predictable, violates IND-CPA (send $M_1 \oplus IV_1 \oplus IV_2$)

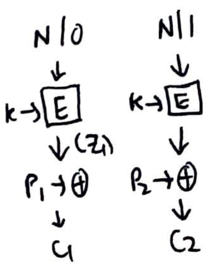
3. Counter Mode (CTR)

   $Z_i = E_k(IV + i)$

   $C_i = Z_i \oplus M$

   $Enc(k, P_1 | P_2 | \cdots)$

   $= N | C_1 | C_2 \cdots$



- Nonce is public
- No computational dependency btw rounds
- IV is replaced with Nonce and counter
- IV can be known ahead of time

CBC compromise less or = CTR if IV same in 2 MSG

*encryptions preserve length

Assymetric Key Cryptography

P_k = public key

S_k = secret key

ONE WAY FUNCTIONS

1) easy to compute $f(x)$
2) given $f(x)$, hard to compute x

Discrete Log Problem

$f(x) = g^x \bmod p$: p prime (2048 bits), g random value in $(2, p-1)$

assumption: $f(x)$ is ONE WAY

*g^x is easy to compute w/ Repeated Squaring

Diffie-Hellman Key Exchange

public : prime p, g between $(2, p-1)$

secret : a, b

public : $A = g^a \bmod p$, $B = g^b \bmod p$

key $K = g^{ab} \bmod p$

becomes symmetric key

El Gamal encryption

Public: prime p, $g \in [2, p-1]$, $PK = g^k \bmod p$

Secret: $k \in [2, p-2]$

To send encryption :

$Enc(P_k, m)$: pick \$ R,

$C = (g^r \bmod p \,|\, m \cdot \underbrace{P_k^{\,r} \bmod p}_{g^{kr} \bmod p})$

To decrypt: take $g^r \bmod p$, raise it to k, divide it to get m.

Padding

add 1 then 0s.

Man in the Middle Attack

C stuff

size() - how much space allocated by pointer

strlen()

sizeof()

CS 161 Computer Security MT2 + Finals

4. Message Authentication Code and Digital Signatures

	Symmetric-key	Assymetric Key
Confidentiality	Sym-Key Enc. (AES-CBC)	Public Key enc. (El Gamal, RSA Enc)
Integrity and Authentication	MACs (AES-CBC-MAC)	Digital Signature (RSA signature)

- MAC is computed F(Key, Message) and tagged on to a message, or ciphertext. The key is shared and private and the recipient then checks if its correct
- Guarantees Integrity/Aunthentication

Hash Functions (Cryptographic)

- "fingerprint" of a message

1) • ONE WAY - given x, easy to compute $H(x)$. Given $H(x)$, infeasible to find x. (preimage resistant)

2) • SECOND PREIMAGE RESISTANT - given message x, infeasible to find x' s.t $x \neq x'$ and $H(x) = H'(x')$

3) • COLLISION RESISTANT - infeasible to find any x and x' s.t $H(x) = H'(x')$

5. Digital Signatures

- public key version of a MAC
- public key (verification key), and private key (signing key).

ex) Alice has public key (verification key), and private key (signing key). Bob verify w/ public key. Alice signs message and sends to Bob.

- Key Generation - Randomized algorithm KeyGen to make public, private key
- Sign $(k, m) = S$
- Verify$_u$ (M, S) true if check out

Key Management

How to manage keys? How does Alice key is found out by Bob

Digital Certificates:

A piece of info presenting someones publickey, signed by a someone. The someone should be trusted.

Public Key Infrastructure and Hierarchial

def: CA : Certificate Authority a party who issues certificates. Browser has hardcoded CA's. publickey

Certificate Chain

Tree of chains of certification: Trust root, and follow. Allows distribution

Revocation

- What happens if bad certificates are given out?
  1) Expiration date
  2) Revocation List

6. Passwords

- password salt + hash it

7. Networking: Internet

OSI - 7 layer model : Internet Layering

7. Application - human readable content, HTML, email
4. Transport - TCP/UDP: creates end-to-end connection
3. Network - finds routes through internet to actually send msg. IP address protocol
2. Link - breaks down routes in the network layer into hops between networks.
1. Physical - individual bits w/ physical protocols

def: offpath adversaries : cannot read or modify msg over a connection

def: onpath adversaries : can read but not modify msgs

def: inpath adversaries : can read, modify, block msgs. MITM

Lower Layers 2: ARP, DHCP

- common link layer: ethernet : assigns 6 byte MAC address to each computer on the LAN.
  - Ethernet : broadcast only - each node hears messages from all other nodes.

ARP : Address Resolution Protocol

- Translate Global IP address → MAC address

LAN : if Alice sends to Bob and knows IP is 1.1.1.1, then Alice broadcast to LAN, who is 1.1.1.1.
- Bob responds my MAC is ___.

NON LAN : gateway responds instead of Bob.

DHCP: Dynamic Host Config Protocol

- handles setup when a computer first joins a network.

You Need
1. IP address - so ppl can contact you
2. IP address of DNS server - so you can translate URLs
3. IP address of gateway - to contact internet

Protocol
1. Client Discover - client broadcasts a request for config
2. Server offer - Any server able to offer IP addresses responds w/ config settings
3. Client Request - client broadcast which it chose
4. server Acknowledgement - server confirmation

Layer 3: IP (Network Layer)

Internet Protocol is layer 3. Connect network of networks
- provides means of transferring variable-length network packets to a host via networks
- Routers, Gateways.

Layer 4: TCP/UDP

TCP: Transmission Control Protocol
- Reliable
- inorder
- connection based stream protocol
- TCP connections identified by 5 tuple (Client IP, Client Port, Server IP, S.P., Protocol = TCP)

Protocol
1. client sends TCP SYN to server (seq = x)
2. server sends SYN (seq = y, ack = x+1) ACK
3. Client sends ACK (ack = y+1)

- No integrity + confidentiality

UDP : user datagram protocol

Fast, less secure, no guarantee version of TCP. but no ID #

Layer 6.5 TLS (Transport Layer Security) (SSL)

Integrity + Confidentiality
- end to end security in communication.
- HTTPS, SMTP use TLS.
- Built on top of TCP. Extension of it. Starts after 3.

4. Client sends Random # R_B and list of ENC protocols
5. server sends Random # R_S, selected Enc protocol, server's certificate, (publickey + CA signature)
   ↑ (server has corresponding private key)

— Generating Premaster Secret —
1. RSA: client generates random PS and encrypt it w/ server public key.
2. Diffie Hellman - after 5, server sends $\{g, p, g^a \bmod p\}$ + signature. client sends $g^b \bmod p$

6. SEND MAC of dialog, and with the PS, generate 4 shared symmetric keys.

Both know: C_B, I_B, C_S, I_S | use these to MAC after step
enc, integ, enc, integ | 5. to ensure integrity + confidentiality

TLS
Replay attacks Not valid be of Random generated #s.

DNS (Domain Name System Server)
- translate domain name to IP address

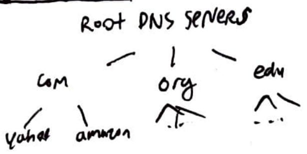
DNS Message Query
- every website starts w/ DNS lookup. uses UDP.

Query

Random ID | isQuery | isSuccess?
↳ NOERROR
NXDomain

Servers
- Computers delegate lookup to DNS to DNS recursive Resolver.

1. DNS Stub Resolver on computer sends query to Recursive resolver
   - configured by ISP or DHCP configuration
2. DNS Authority Servers - answers the queries

DNS Lookup


1. ask one of Root servers

DNS Security
Baliwick
- attackers can only provide name server to provide records under its domain.
  ex) berkeley.edu can only give berkeley.edu

On path Attacker
- Totally insecure: attacker can fill in ID and race to WIN.

Off path attacker
- attacker has $1/2^{16}$ chance to guess Random ID.
- Kaminsky Attack
  - relies on querying for Nonexistant domains
  - since NX Status, nothing is cached, attacker can do attacker.berkeley.edu, and attach malicious Additional section to cache the malicious additional info.
  - trick user to send multiple queries for NX domain.

DNSSEC
- provides integrity & authentication
- designed as PKI
- chain of sign keys: starting at Root

Two new fields: DS, RRSIG

DS ↓ public key of next Name server

RRSIG ↓ Signature of public key of DS record ↑ given as the proof

check this

- practicality: name servers precompute signatures on ranges of NONexistent domains.

8. Network Intrusion Detection (NIDS)
NIDS - cheaper, easier to implement, chokepoint. can't work w/ HTTPS

HIDS - roll out on each application. expensive. works in backend.

Logging - just log all incoming HTTPs.

Detection Technique
1. Signature based: basically ctrl-F, against publicly known attacks
2. Anomaly Based: ML. define what is normal
3. Behavior Based: flag down every behavior that has a characteristic of an attack.
4. Logging: Efficiency: 24 after attack.

9. Web Security
Layer 7: HTTP (application level): common data communication protocol.

def: Frames: embedding a page within a page
1. Outer page can only specify sizing and placement of frame
2. outer page cannot change contents of inner page. Vice Versa

SAME ORIGIN POLICY (SOP)
1. each site in browser is isolated from others

def: ORIGIN: (protocol, hostname, port): uses string matching.

10. XSS Attack
- attack happens within same origin

Type 1: Stored XSS: attacker leaves JS lying around on benign webservice for victim to load.

key trick: server fails to ensure that content uploaded to server does not contain embedded scripts

ex) post on twitter embedded JS to automatically make everyone Retweet it.

Type 2: Reflected XSS:
- attacker gets the victim to visit URL for bank.com that embeds malicious javascript.
- server echoes it back to victim user in Response
- server executes the script within same origin
- victims browser executes the script within same origin as bank.com

key trick: server fails to ensure that output it generates does not contain embedded scripts.

ex) URL: bank.com/search=<script?c/...>
server responds back w/ HTML with the script in it, and then the script can send sensitive info to other evil.com,

Fixes: validate input/output, use Content Security Policy (CSP)

11. Session Management
Cookies:
- created when first visit a website.
- can store preferences or login
- stored directly in browser

Cookie: Name = VALUE, domain, path, secure

cookie policy:
1. what scopes a URL-host name web server is allowed to set on a cookie
   - browser checks if web server can set the cookie, if not then reject.

JS can't touch ↑
- HTTPOnly
- Secure (only HTTPS)

- Domain: any domain suffix of URL-Hostname except TLD.
  ex login.site.com ⇒ login.site.com, .site.com
- path: can be set to anything

2. When browser sends a cookie
   - sends all cookies in URL scope
     - cookie domain is domain suffix of URL domain
     - cookie path is prefix of URL path and
     - HTTPS if cookie is secure

ex) A cookie w/ domain example.com sent to foo.example.com
- sends all cookies

Session Management

Session Token - temporary identifier for user

- if an attacker gets a session token, it could access the user's account for the duration of that token.

ex) BROWSER — Web Site

GET/index.html →

← set anonymous session token

GET /books.html → anonymous session token

POST/ do-login → username/ password

← elevate to loggedIN session token

Post/ purchase → logged-in session token

- stored in server database
- created based on cookie

Session using cookies

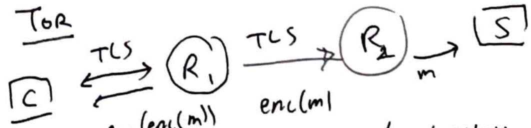
Browser — SERVER

POST / login.cgi →

← Set cookie : session token

GET/POST → cookie: session token

Tor



1. perform TLS handshake w/ each relay
2. Encrypt request in layers of relay sym. keys.

CSRF (Cross Site Request Forgery)

ex)
- user log into bank.com
  - session cookie remains in browser state
- user visits malicious site containing:
  - go to bank.com and send money to me.
- browser authenticates that w/ token, auto attached.

- works for forms
- takes advantage of ~~tokens~~ session token

Defense

1. CSRF token
   - hidden value that you get and put on any form,

2. Referer Validation Defense
   - when browser issues an HTTP request, it includes referer header that indicates which URL initiated the request.

- sessions can be hijacked through packet sniffing, so HTTPS it.

UI Attacks

clickjacking attacks - exploitation where user's mouse click is used in a way that was not intended by user. Attack on perception on UI

Simple attacks : can do window.open(attack.com) but show diff URL

More complex : using Frames

- clickjacking - bypass SOP by overlaying certain UI block over the frame
- cursorjacking - lay a fake visible cursor at a certain distance and it'll click elsewhere

Defense

1. user confirmation
2. UI Randomization
3. Framebusting
4. X-frames options for browser