Chapter 10

Monday, August 28, 2023 11:36 AM

Introducing Cryptography Concepts

Core concepts

- **Integrity** - data was not been modified

hashing

hash - # derived from performing calc on data

create fixed-length string of bits or hex chars which can't be reversed

common algo is **secure hashing algorithm** } **SHA-**

- **Confidentiality** - data only viewable by authorized users

encryption - scrambles data to make it unreadable if intercept

algo + key

algo + key

Symmetric encryption - use same key to encrypt/decrypt

block or stream cipher

Stream cipher - encrypt data 1 bit at a time

block cipher - encrypt in blocks

Asymmetric encryption - uses two keys pub/priv

requires PKI to issue certificates

anything encrypted by pub can only be decrypted w/ priv

vice versa

- Steganography - hides data in other files

ex: inside white space of image

- digital signatures provide

- digital signatures provide
  - authentication
  - non-repudiation - can't deny action
  - integrity

- sign emails w/ digital signature
  hash of email encrypted w/
  senders priv key

  only senders pub key can decrypt

Providing Integrity with Hashing

no matter how many times you do hash on data, it will be same if data is unchanged

SHA-3 is calc # in hex

SHA-3 256 is 64 hex

Hash versus Checksum

hash is longer and used in crypto implementations

small piece of data

Checksum - a small piece of data used to quickly verify integrity of data

1 or 2 bits

Ex: RAID-5 use a single parity bit per byte

MD5

Message Digest 5 MD5 - common hashing algo that produces 128-bit hash

32 hex

no longer secure

sometimes used to verify quick checksum

Secure Hash Algorithms

SHA - group of hash algos w/ ... in grouped four families

SHA - group of

variations in grouped four functions

SHA-0 - not used

SHA-1 - updated version that creates

160-bit hashes

like MD5

no longer secure

SHA-2

SHA-256

SHA-512

SHA-224 = truncated 256

SHA-384 = truncated 512

SHA-3

alt to SHA-2

Not created by NSA

has same hash sizes as SHA-2

HMAC

hash-based message authentication code

HMAC - fixed-length string of

hash-based message
HMAC - fixed-length string of bits similar to MD5 / sha256

uses shared secret key to add randomness

only sender / receiver know

ex: using MD5 to send message

- calc hash
- use secret key to compute another calc on hash

HMAC-MD5 is secure if secret key is long enough

internet protocol security IPsec and TLS often use HMAC

Hashing Files

sha256sum = free prog to calc hashes

hashes will always be same if data

does

Hashes are 1-way

Hashing Messages

does not encrypt but provides integrity

Using HMAC

if attacker can modify message in transit, they may be able to alter hash to trick into thinking their hash is correct

HMAC solves this ↖

only sender/receiver know shared secret key

they use key to make HMAC-MD5 hash out of sent MD5 hash

Hashing Passwords

passwords aren't stored or sent over cleartext, their hashes are

many tools to discover weak pwords

Sometimes even SHA-3 can be cracked w/out salting

Understanding Hash Collisions

occurs when diff inputs have same hash

MD5 highly susceptible

Understanding Password Attacks

discover or bypass pwords

online password attack - discover pword from online system

... to login to acc

ex: repeatedly try to login to acc

nrack = tool to crack pword
w/ brute force

offline password attack - discover
pword from captured db or
packet scan

common indicator of online is
in system logs

Windows security log / Event viewer
Event ID 4625
failed login

Event ID 4740
locked account

Dictionary Attacks
uses dict of words and attempts
[illegible] as pword

uses dict of words
every word as pword

thwarted by strong pwords

Brute Force Attacks

attempt to guess all possible char combos

thwart w/ strong pwords and store in encrypt/hash format

Spraying Attacks

Special brute force to avoid lock outs

- try pword w/ large list of unames
- try the same w/ next pword

avoids lockouts because it takes a while to get back to 1st acc

Pass the Hash Attacks

attacker discovers hash of pword
[illegible] it to log on system

attacker discovers
then uses it to log on system
as user

any auth proto that passes hash
over network unencrypted
is vuln

when attackers gain admin access
to system, they try to
steal hashes stored in multiple
locations on the pc

- security accounts manager SAM
- Local sec authority subsystem LSASS
- Credential Manager CredMan
- LSA Secrets

common indicator is usage of
NTLM as the auth package
or a logon process of NtLmSSP
Event ID 4624

or a logon process
shown in Event ID 4624

can be correlated w/ EID 4672
to determine privileges used
w/ connection

normal users wouldnt use admin
privileges when connecting to
other pcs

Birtuday Attacks

attacker attempts to create pword
w/ same hash as user's

thwarted by increasing # of bits
to create hash

ex: SHA-512

Rainbow Table Attacks

attempt to discover pword from
hash

rainbow table - huge db of possible
hashed hashes

rainbow table - huge ...
password w/ precomputed hashes

W/out rainbow table you could:

1. guess a password
2. hash guess
3. compare hash
4. repeat

Usually offline, compare hashes in db to hashes in rainbow table

Salting Passwords

salt - random data such as 2 chars added to password before hashing

thwarts
- rainbow table
- brute force
- dictionary

Key Stretching

advanced technique to increase strength of stored passwords

adds crypto stretching algo to salted password

Common stretching techniques

- Bcrypt
  - used to protect password file on Unix/Linux
  - Blowfish block cipher
  - salts password and encrypts it. Now can do multiple times
  - 60 char string
  - app runs bcrypt on given password and compares w stored bcrypt password
  - nd scrypt - another set

can add pepper - another set of random bits

- Password-based key derivation function 2 PBKDF2

uses salts of at least 64 bits and uses pseudo random function like HMAC to protect passwords

can go through process 1mil times

size of hash can vary 128 bit, 256, 512

weakness is it can be configed to use less compute time makes it easier for attackers to guess

- Argon2

alt key stretching method

user salt and pass through

algo several times

Providing Confidentiality with Encryption

encryption scrambles data creating

ciphertext

Data at rest

Stored on media

Data in transit

sent over net

Data in processing/use

used by a pc

not encrypted

if it is, app will decrypt

and store in mem

encrypt again

and store

if change data, encrypt again before storing again

encryption method include

- algo
- key - # that provides variability for the encryption

Symmetric Encryption

uses same key

Substitution cipher - replace plaintext with cipher using fixed system

ex: replace char with char that comes after it alphabetically

ROT13 key of 13

not encryption, just obfuscate

Sophisticated symm use complex algos

n-bit keys or

sophisticated

AES uses 128-bit keys or more

Change keys often

ex: use diff keys for diff files

Block versus Stream Ciphers

sym algo use either block or stream ciper

block ciper

specific-sized blocks

64-bit, 128-bit, ...

Stream ciper

encrypts data as stream of bits

much more efficient when size of data is unknown

block more efficient

block more efficient when known

key can never be reused w/ stream

Common Symetric Algoritums

Advanced Encryption Standard AES

strong block cipher

128-bit blocks

keys of 128-bit, 192, 256

strengths:

- Fast
  - 2 pass to en/decrypt
- Efficient
  - less resource intensive
- Strong

3DES

mock cipher

3DES

block cipher

Data encryption standard DES

3 separate passes to encrypt

64-bit blocks

used when hardware can't support AES

key size 56, 112, 168

Blowfish and Twofish

blowfish

block cipher

widely used today

64-bit blocks

32-448 keys

can be faster than AES

twofish

similar but encrypts in 128-bit blocks

128, 192, 256 bit keys

Asymmetric Encryption

2 keys in pair

- only matching pub/priv key can decrypt
- never share priv
- share pub, embed in shared certificate

Strong but resource intensive

key Exchange

Crypto method to share crypto keys between two entities

asymm uses key exchange to share a symmetric key

most uses of asymm are for key exchange

Certificates

key element of asymm

certificate - digital document that includes pub key and info about owner of cert

issued by CA

also used for auth and digital sigs

users and apps share cert file to share public key

Common elements

ids cert

Common elements:

- Serial # - uniquely ids cert
  CA uses to validate
  if revoke, publishes in certificate revokation list CRL

- issuer - CA

- Validity dates
  valid from-to

- Subject - owner of cert
  ex: google

- public key

- usage
  some are only for encryption or authentication

Distinguished Name Attributes (CA)

- CN - common name

- O - Organization
- L - Locality
- S - State or province
- C - Country

Ephemeral keys

has short lifetime and is recreated for each session

Static key - semipermanent and stays the same for long time

Certificates have static keys that exist for duration of cert

perfect forward secrecy

ephemeral keys comply with in asymm

crypto system generates random public keys for each session

compromise of key does not compromise any past keys

Elliptic Curve Cryptography ECC

doesnt take as much processing power as other crypto methods

uses math to form elliptical curve graphs points on curve to create keys

keys can be much smaller

Digital Signature Algorithm DSA

uses key pairs managed by PKI with pub key dist in a cert

Elliptical Curve DSA ECDSA

can also be used for digital sigs

sigs

256-bit EC public key

=

3072-bit DSA key

good for low power devices

Quantum Computing

Quantum cryptography - uses quantum mechanical properties to perform crypto tasks

regular computing bits are 1 or 0

quantum uses quantum bits qubits - can be 2 values at the same time

Superposition allows qubits to be superposed (added together)

2nd quantum state

qubits can be entangled and creates

qubits can be entangled and
dependency between them

cant copy data in quantum
state

quantum communication uses
no cloning to provide integrity
if attackers intercept, it
changes

Quantum Cryptography

quantum key distribution QKD
allows two parties to establish
shared key
like how asymm is used for
symm key gen

no cloning protects key gen

Post-Quantum Cryptography
quantum-proof or quantum resistant

Post-quantum [illegible]

quantum-proof or quantum resistant

crypto algos that are likely to be resistant to attacks using a quantum pc

NIST call for proposals

for quantum-resistant public key crypto algos

Lightweight Cryptography

crypto deployed to small devices like RFID tags

ISO 29192-2:2019 block ciphers

- Present 64 bit block 80/128 key
- CLEFIA 128 block 128/192/256 key
- 2EA 128 block 128/192/256 key

Homomorphic Encryption

Homomorphic Encryption

allows data to remain encrypted while in use

useful for situations like health care so it can be shared and updated in real time

Key Length

algo always stays the same

to strengthen, need to increase key length

RSA - primary public key crypto algo

key size 1024, 2048, 4096

Modes of Operation

authenticated encryption modes ensure confi of data and

ensure con..
authenticity

often used with symm block
ciphers

commonly combine symm algo
w/ message authentication code
MAC

auth performed on each block

Same as using HMAC, first
key to decrypt and other
is used to calc MAC to
prove it came from other party

Counter mode cipher CTR - converts
block cipher into stream

combines initialization vector IV
with counter and user result
to encrypt block

to encrypt block

IV = fixed-size random/pseudo random #

Authenticated encryption

confidentiality and authenticity

Counter mode

form of auth encryption

allows block to function as stream

Unauthenticated

confidentiality but not auth

each block in CTR uses same IV but CTR uses counter for diff encrypt for each block

CTR widely used

IV and counter are known between

parties

Steganography

hides data inside other data

hashing detects steganography

audio, image, video

Audio Steganography

humans can't hear 18khz - 20khz

audio beacons to id user activity

Image steganography

manip bits of file or by hiding in white space

can modify least significant bit

ex:

red = 255, 0, 0

[illegible] 255 to 254

change 255 to 254
wont be recognized

using whitespace can sometimes
make it possible to not
change file size

Video Steganography

also modify least sig bit

but can sometimes cause audio
noise

Using Cryptographic Protocols

pub/priv - which one encrypts?
depends on use case

- Email digsigs

sender priv key encrypts
pub key decrypts

Sender pub key decrypts

- Email Encryption

  recip pub key encrypts

  recip priv key decrypts

- Website Encryption

  web site pub key encrypts

  web site priv key decrypts

  Symmetric key encrypts data in session

Email and website commonly use symm + asymm

asym for keyex

symm for data encryption

Protecting Email

either digital sig or encryption

Signing Email with Digital Signatures

Signing Email with Digital Sig

Digital signature Algorithm DSA

uses encrypted hash of message

hash is encrypted with sender's private key

if recipient can decrypt hash:

- Authentication
  know that it came from sender
- Non-repudiation
  sender can't deny that they sent it
- Integrity
  message has not been modded

Digital sigs need:

- hashing
  hash the message

- Certificates
  include sender info
- public/private keys
  public key often sent in S/MIME.p7s file

note that for basic use that only the hash of the message is encrypted

message can be too but not in this scenario

Senders pub key is in cert

why not just encrypt message too?

resources, especially for long email + attach

Encrypting Email

want to ensure email is only

Want to ensure email is only read by authorized users

Encrypting Email with Only Asymmetric Encryption

1. Get copy of recip cert w/ pub key
2. Encrypt email w/ pub key
3. Send email
4. Recip decrypts w/ priv key

Encrypting Email with Asymmetric and Symmetric Encryption

asymm slow, symm fast

most email use asymm to share session key

Symm to encrypt data w/ session key

1. Sender ids symm key to encrypt email

2. Encrypt email

3. Get copy of recip pub key and cert

4. Use recip pub key to encrypt the symm key

5. Send encrypted email and encrypted symm key

6. Recip decrypts symm key with priv key

7. Recip decrypts email using symm key

S/MIME

Secure / Multipurpose Internet Mail

Secure / Multipurpose Internet Mail Extensions S/MIME

one of most pop standards for digitally sign/encrypt email

both asymm/symm

can encrypt at rest and transit

uses cryptographic Message Syntax CMS -
allows it to use variety of hash/encrypt algos

typically use these ports

- 495 for POP3 over TLS
- 587 for SMTP over TLS
- 993 for IMAP over TLS

HTTPS Transport Encryption

transport encryption encrypt data in transit

in transit

both over internet and internal net

TLS versus SSL

SSL out of date but basically SSL = TLS or SSL/TLS

common to encrypt HTTPS with TLS

can also be used to encrypt others like FTPS

TLS provides cert based authenticate

encrypts w/ both asymm/symm

Requires certs from CA internal/external

Encrypting HTTPS Traffic with TLS

Symm used to encrypt data

Client and server must know

Client and server must ...
what symm key is

TLS HTTPS process

1. client begins by requesting an HTTPS session
   click link, search URL, ...
2. Server responds by sending server's cert which has pub key
3. Client creates symmetric key and encrypts with servers pub key
   Symm key also called session key
4. Client sends encrypted session key, which only server can decrypt

priv can decrypt

5. Server decrypts session key

6. All session data is encrypted w/ session key

Downgrade Attacks on Weak Implement

downgrade attack - forces sys to downgrade sec

often associated w/ crypto attacks due to weak imp of cipher suites

EX: server supports SSL and TLS

attacker modifies pc to use w/ SSL

can then launch SSL attacks

Padding oracle on downgrading

Known Encryption POODLE

Legacy Encryption POODLE attack

Need to disable SSL and weak cipher suites

Blockchain

distributed, decentralised, public ledger

Each block has:

- info about trans
  - date, time, amount
- info about parties
  - digisig
- unique hash

Block is added after:

- transaction occurs
- verified by net of pc
  - ...ed in block

- verified by
- accurately recorded in block
- block given unique hash

block also has hash of most recent block

Crypto Diversity

use different methods to protect security keys

ex: using multiple HSMs to protect keys

Identifying Limitations

diff algos have diff limits

Resource versus security Constraints

more data = more encrypt time and resources

also more process power to decrypt

Speed and Time

Speed and Time

Speed = how long algo takes

want quick for encrypt/decrypt

slow for hashing

Size and Computational Overhead

Size = how much mem space algo needs

mostly only matters for small devices

Entropy

randomness of a crypto algo

more = more sec

Predictability

associated w/ random # gen

used to make encryption keys

pseudo-random given same input will give same output

will give same output

Weak keys

short or small

at least 2048

Longevity

how long you expect to use algo

ex: upping RSA key bits over time

Reuse

Symm = keys shouldnt be reused

especially stream ciphers

Plaintext Attack

possible if attacker has known plaintext data and the ciphertext created from it

can use to discover en/decrypt

can use to discover en/decrypt methods

Chosen plaintext attack

attacker doesnt have all ciphertext

decrypt desired part of encrypted text to find out rest

Ciphertext only attack

has no plaintext

typically only work on weak encryption

Common use cases

- Supporting Integrity

  hashing

- Supporting Confidentiality

  encryption

- Non-repudiation

  digisigs

digisigs

- High Resiliency
  sec of encryption key even if part of key is discovered

- obfuscation
  steganography

- Low power devices
  ECC and light weight crypto

- Low latency
  online certificate Status protocol
  OCSP validates cert in real time

  CRLs are cached and not realtime

Exploring PKI components

public key infrastructure PKI - group

public key infrastructure

of tech to request, create, manage, store, distribute, and revoke digital certs

allows 2 entities to comm securely w/out knowing each other

comm securely through insecure net

Certificate Authority

CA issues, manages, validate, revokes certs

based on trust

why would pc trust CA?

trust paths

Certificate Trust Models

CAs are trusted by placing copy of their root cert into a trusted root CA store

root certificate - first cert created by CA that ids it

Centralized trust model

Hierarchal trust model

pub CA creates first CA (root)

can then create intermediate CA and child CAs

root
↓
int
↓
child

certificate chaining - combines all CA from root down

Registration Authority and CSRs

Process to getting certificate

- create pub/priv key
- certificate signing request

- Create certificate signing request
  CSR
  purpose of cert
  info about site
  pub key
  most need to be in public-key crypto standards PKCS #10

- CA verifies your ID and creates cert
- registers cert w/ website

registration authority RA - can assist CA by collecting registration info

Online versus offline CAs

Online = access over internet
can submit CSR by auto process

offline can only submit manually

typically root is offline and
inter/child can be online

Updating and Revoking Certificates

certs expire from valid to-from dates

CA can revoke compromised certs

Certificate Revocation List

CRL is version 2 cert that holds
revoked cert serial #s

Validating a Certificate

make sure that

- not expired
- CA is trusted
- cert not revoked

1. Client init HTTPS sesh
2. Server responds w/ copy of cert
   & for CRL

2. Server responds w/ cert

3. Client queries CA for CRL

4. CA responds w/ CRL

Online certificate status protocol OCSP

allows client to query CA w/ serial #

OCSP stapling - cert presenter obtains timestamped ocsp response from CA

CA signs it

appends to cert

removes need for clients to query

Public Key Pinning

prevent attackers from impersonate site w/ fraud certs

is to HTTPS w/

Client responds to HTTPS w/ extra header

includes list of hashes derived from valid pub keys used by site

also max-age field

When client connects again, it recalcs hashes and compares w/ stored

Hashes are of one or more certs used by site

Must also include backup key

Key Escrow

placing a copy of private key in safe env

[illegible] 3rd party, ...

employees, 3rd party, ...

key recovery agent

Bitlocker

2 key for a-to-do

second key only used by recovery agent

key Management

all steps taken to manage public and priv keys in PKI

- keep priv keys priv
- dist pub keys
- revoke certs

Comparing Certificate Types

Machine/computer

issued to device or pc

id pc in domain

User

used for encryption, auth, ---

Email

encrypt email and digicerts

Code Signing

validate auth of exe apps/scripts

Self-signed

issued by private CA

must be placed in trusted root store for systems

Root

Wildcard

Start w/ *

can be used for mult domains as long as all have same

as long as all

root domain

*.google.com

Subject alternative name SAN

mult domains w/ diff names

under same org

*.google *.android

google.com google.net

Domain Validation

indicates that cert requestor has

control over DNS domain

CA takes extra steps to verify

Extended Validation EV

use add steps to domain valid

Comparing Certificate Formats

most use one of X.509 v3

rel use

most use one -

Certs used to distribute CRL use X.509 v2

Stored as binary file or as BASE64 ASCII files

Base format of certs is Canonical Encoding Rules CER or Distinguishing Encoding Rules DER

CER = ASCII

DER = Binary

Some have headers like

----- Begin cert -----

Privacy Enhanced Mail PEM

not just email

CER or DER

CER or DER

doesnt only use .pem

P7B

Use PKCS#7

CER based

used to share pub keys

P12

PKCS #12

DER

used to hold certs w/ priv key

Personal Information Exchange PFX

predecessor to P12

Same usage

Assessment Notes

1. Hashes of updates is for integrity

1. Hashes of updates is for integrity

2. Hashing gives integrity for signed emails

3. Spraying Attack = intervals of failed logons

4. Salt decreases chances of attacker discover hashed passwords

5. Stream cipher encrypts bits at a time

both used for symmetric

6. Low processing power encrypt/decrypt

Elliptical Curve crypto ECC

7. CSR = request cert for server

8. Verify trust certs are valid

CA is down

8. Verify [illegible]
even if pub CA is down

CRL

private CA can't validate
certs from pub CA

9. Realtime alt to CRL = OCSP

10. Public key pinning prevents
attackers from impersonate
site w/ fake certs

11. Digital signatures support
non-repudiation

12. Prevent someone from denying
accountability = Non-repudiation

13. Digital sigs give authentication

[illegible] for multi domain/

14. SAN = cert for multi domain w/ different names

Subject Alternative Name

15. Private key stored in key escrow