# Computer Security

## 2019 Exam 3 Review

Paul Krzyzanowski

Rutgers University

Spring 2019

# Question 1 version B: 2, C: 3

*Hashed* passwords:

(a) Slow down the login validation process, making guessing more time-consuming.

(b) Make it difficult to crack a password even when presented with its hash.

(c) Ensure that an attacker will not be able to tell if two users have identical passwords.

(d) Speed up the login validation process, since only hashed values need to be compared.

---

Hashes are not invertible.

# Question 2

version B: 3, C: 1

*Salt* in a password:

(a) Will ensure that an attacker will not be able to tell if two users have identical passwords.

(b) Slows down the login validation process, making guessing more time-consuming.

(c) Makes it essentially impossible to decrypt the password.

(d) Serves an encryption key to create combined encryption and hashing.

---

Salt is a random value prepended to a password

$\text{hash}(\text{salt}_0 \mid \text{“password”}) \neq \text{hash}(\text{salt}_1 \mid \text{“password”})$

# Question 3

version B: 1, C: 2

Which is *NOT* a form of two-factor authentication?

(a) Password + Time-based One-time Password (TOTP)

(b) User name + Fingerprint scan

(c) Password + SMS code

(d) Access card + Iris scan

---

User name + fingerprint scan is **single-factor** authentication, just like user name + password

The user name serves as identification, not authentication

# Question 4

version B: 5, C: 6

The *Time-based One-Time Password* (TOTP) relies on:

(a) Both sides having a shared secret value.

(b) Having the client encrypt a challenge sent by the server using a time-based function.

(c) The server controlling the time of day during which a user is allowed to log in.

(d) Providing authentication for services that a user will use only one time.

---

Password = f(secret, time)

There is no challenge sent by the server

# Question 5 version B: 6, C: 7

*Universal Second Factor* (U2F) authentication is best described as:

(a) A combined authentication and key exchange protocol.

(b) A one-time password generating device that generates a password based on the server address and time.

(c) A USB-based biometric authentication device that sends encoded biometric data to the server.

(d) A challenge-response protocol based on public keys and digital signatures.

1. Server sends a challenge

2. Browser forwards it to the U2F device

3. U2F device creates a signature of the challenge using its private key

4. Sends it back to the browser, which sends it back to the service

# Question 6

version B: 7, C: 4

A *man-in-the-middle* (MITM) attack can best be avoided by:

(a) Exchanging Diffie-Hellman keys to create a common key to encrypt a session.

(b) Using a time-based one-time password.

(c) Using a public key from the server certificate to send an encrypted session key.

(d) Using hashed passwords.

The man in the middle will not be able to forge a certificate & will not have the private key to decrypt data

(a) The MITM can create another set of D-H keys

(b) The MITM can simply relay the password request/response

(d) These can be relayed too (if needed by some protocol)

# Question 7 version B: 4, C: 5

CAPTCHA works on the principle that:

(a) Certain image recognition problems are much more difficult for computers than humans.

(b) Image recognition tasks cannot be scripted to automate the authentication process.

(c) Even if image recognition is automated, the web interaction still requires a human.

(d) The delay of solving CAPTCHA puzzles is long enough to slow down any botnet-based attacks.

---

The reason CAPTCHA was created was to find tasks that are easy for humans but extremely difficult for computers

# Question 8 version B: 9, C: 10

This attack on the link layer allows an attacker to see traffic on all machines on a local area network:

(a) DHCP (dynamic host configuration protocol) spoofing.
(b) ARP (address resolution protocol) cache poisoning.
(c) DNS (domain name system) rebinding.
(d) CAM (content addressable memory) table overflow.

A switch table overflow will cause all traffic to be relayed onto all switch ports

# Question 9

version B: 10, C: 11

In which attack does a system impersonate a network switch?

(a) ARP cache poisoning.

(b) DHCP spoofing.

(c) VLAN hopping.

(d) CAM table overflow.

VLAN hopping uses switch spoofing –

computer identifies itself as a switch with a trunk connection

Gets 802.1Q Ethernet frames with all VLAN traffic

# Question 10 version B: 11, C: 8

The purpose of *SYN cookies* is to:

(a) Create a shared key for a connection so that the server will accept only properly-signed messages.

(b) Allow the client to authenticate that the server is not an imposter.

(c) Avoid allocating memory for TCP connections until the server gets a response from the client.

(d) Provide a simple way for a client and server to set up an encrypted link.

---

SYN cookies reduce SYN flooding attacks

Respond with a ack#:

*Cookie = hash(src_addr, dest_addr, src_port, dest_port, SECRET)*

When the client sends its ACK back to the server,

recompute the cookie to see if it matches the ACK #

# Question 11

version B: 8, C: 9

A DNS rebinding attack can:

(a) Cause a client to contact a different DNS server.
(b) Make network traffic visible to all systems on the local area network.
(c) Cause a browser to unknowingly violate the same-origin policy.
(d) Enable network messages to bypass firewall filters.

---

Attacker owns the DNS server for some domain, bad.org

JavaScript can interact with bad.org because of the same-origin policy. However, the time-to-live value for bad.org is short.

For future references to bad.org, the OS will do a DNS lookup. The attacker could have changed the DNS server to return the address of a different system (even a local one).

# Question 12 version B: 13, C: 14

The IPSec Authentication Header (AH) protocol ensures that:
(a) Packets are not forged.
(b) Packets are encrypted.
(c) Packets are compressed.
(d) All of the above..

---

- AH protocol
  - Makes communications tamper-proof by adding an HMAC to each message
  - Content is not encrypted
- ESP (Encapsulating Security Payload)
  - Same but adds data encryption

# Question 13 version B: 14, C: 15

Blocking all TCP traffic from any systems whose IP address is in the range 128.8/16 requires, at a minimum:

(a) A screening router with stateful inspection (SPI).
(b) A packet filter.
(c) An Intrusion Prevention System (IPS).
(d) An application proxy.

---

All you need is a packet filter (=screening router)

No need to keep state about past packets

# Question 14

version B: 15, C: 16

A *DMZ* (*demilitarized zone*) is best described as:

(a) A network link that has no systems on it and separates internal and external firewalls.

(b) A network between the Internet and local network that hosts unprotected systems.

(c) A protected subnet hosting systems that provide Internet-facing services.

(d) A highly-secure subnet that is not accessible from the Internet.

---

A DMZ is protected by an exterior router and contains systems hosting Internet-facing services

# Question 15 version B: 16, C: 12

*Amplification* attacks are the result of:

(a) Viruses and worms that propagate at an exponential rate.

(b) A high frequency of packets that overwhelm a firewall, forcing it to pass remaining traffic uninspected.

(c) Messages with spoofed source addresses sent to services that provide large responses.

(d) Social engineering that relies on humans to propagate malware into internal networks.

---

Goal of an DoS amplification attack:

Send *small requests* that create *large responses* that go to the target because the *source address was spoofed*

# Question 16

version B: 12, C: 13

Which technique is most likely to detect *port scanning attacks*?
(a) Protocol-based IDS.
(b) Signature-based IDS.
(c) Anomaly-based IDS.
(d) Packet filter.

Port scanning probes various port numbers on a server to find open ports

(a) It does not follow any protocol – just tries to connect to various ports, often in a random pattern

(b) There is no signature since no data is generally sent

(d) A packet filter cannot detect a pattern of activity

# Question 17 version B: 18, C: 19

*Clickjacking* occurs when:

(a) Malicious JavaScript on a page generates fake mouse click events.
(b) A user is misled into clicking on a page element she did not intend to click.
(c) Mouse clicks are intercepted by a script and their action is changed.
(d) Malware impersonates a browser and sends simulated click events on ads.

---

Clickjacking is an attack where the attacker overlays an image to have the user think that he is clicking some legitimate link or image but is really requesting something else

Can be accomplished by overlaying an invisible frame over an object containing a link

# Question 18 version B: 19, C: 17

Cross-Site Scripting (XSS) occurs when:

(a) JavaScript on a page is permitted to access content on a server different from the origin.

(b) A web page loads JavaScript from a site different from the origin.

(c) User input is not validated and contains a script that is later presented as page content.

(d) A single web page loads multiple JavaScript files from different origins.

---

Cross-site Scripting (XSS) is a code injection attack that allows the attacker to inject client-side scripts into web page

Nothing to do with getting content from different origins.

# Question 19 version B: 17, C: 18

You can reduce the likelihood of Cross-Site Request Forgery (XSRF) by:

(a) Ensuring that commands are not present as parameters in the link.

(b) Validating that the domain in a link is referencing a legitimate service rather than a spoofed one.

(c) Making HTTPS instead of HTTP requests.

(d) Making sure that user input does not contain any JavaScript.

---

Cross-site request forgery is an attack that sends unauthorized requests from a user that the web server trusts.

User will have cookies that authenticate a session on good.org.

Attacker will create a link that directs the user to good.org with some parameters:

```
good.org?action=leave_feeback&org=bad.org&rating=awesome
```

The user will click on this and cause an action to be taken

# Question 20 version B: 21, C: 22

A bitcoin *wallet* needs to store your:

(a) Your list of transactions.
(b) Your public & private keys.
(c) Your account balance.
(d) All of the above.

A wallet stores your public & private keys (one for each of your identities)

The private key is the only secret component that needs to be safeguarded

Account info is based on past transactions in the blockchain: each transaction must point to previous transactions that contain enough coins

# Question 21 version B: 22, C: 23

Bitcoin does *NOT* use:
(a) Digital signatures.
(b) Merkle trees.
(c) Encryption.
(d) Hash functions.

---

Digital signatures: transactions

Merkle trees: transaction storage per block

Hash functions: Blockchain

Encryption: no data is encrypted
(although you may encrypt content in your wallet – but that's outside the functionality of bitcoin)

# Question 22

version B: 23, C: 20

A *proof of work* ensures that:

(a) The integrity of each transaction is rigorously validated by the network of peers.
(b) A majority of systems in the bitcoin network approves each transaction.
(c) Each transaction created by a user is properly signed.
(d) An attacker cannot modify data in a block quickly in a way that yields valid block hashes.

---

PoW is a search for a value that causes the hash(block) to have a specific property.

It takes a lot of work.

Goal: an attacker cannot go back in the chain and rewrite history

# Question 23

version B: 20, C: 21

A *51% attack* means that an attacker:

(a) Modified transactions on over 50% of the nodes.

(b) Caused over 50% of the systems denied transaction.

(c) Harnessed over 50% of the total hashing power.

(d) Took down over 50% of the systems in the bitcoin network by a DoS attack.

This is where cryptocurrencies break.

If someone can harness >50% of total hashing power, they can rewrite history by ensuring that they can create the longest blockchain in the system.

# Question 24

version B: 25, C: 24

Android keeps applications from accessing each other's data by:
(a) Running each process under a different user ID.
(b) Placing each application in a separate container.
(c) Running a process in its own chroot jail.
(d) Running each app in a separate TrustZone instance.

---

Android partitions apps by different Linux user IDs

# Question 25 version B: 24, C: 25

*TrustZone* on Android and *Secure Enclave* on Apple:

(a) Perform run-time validation of executing software.

(b) Runs each app in its own sandbox.

(c) Enable sections of an application to be tagged as security-critical and run at a high priority.

(d) Run security-critical services under a separate operating system.

Separate execution entities that run a separate OS and security-critical services & storage

# The end