

“Ransomware is the greatest thing to ever happen to computer security, all thanks to anarchist crypto nerds who wanted magic internet money.” -SecuriTay

Problem 1 True or false**(46 points)**

Circle True or False. Do not justify your answer.

- (a) TRUE or FALSE: The same origin policy allows `https://www.amazon.com` to access the resources of `http://www.amazon.com`.
- (b) TRUE or FALSE: Framebusting is a defense against cookie tracking.
- (c) TRUE or FALSE: The `httpOnly` flag of a cookie mitigates XSS attacks because it ensures the browser sends the cookie only over `https`.
- (d) TRUE or FALSE: Javascript running on the outer frame of a website can always access the resources of an inner frame in the page.
- (e) TRUE or FALSE: Two Javascript scripts embedded in pages running in two different tabs on a user's browser can never access the resources of each other.
- (f) TRUE or FALSE: All CA-s in your browser need to be trustworthy in order to be protected via TLS (assume no certificate pinning).
- (g) TRUE or FALSE: If you obtained the long-term private key of a party in a TLS connection, you can decrypt past TLS connections if the server uses DHE for the key exchange.
- (h) TRUE or FALSE: TLS does not hide who the communicating parties are from an observer.
- (i) TRUE or FALSE: AES-ECB mode (encrypting each block separately with the block cipher and the same IV) is IND-CPA.
- (j) TRUE or FALSE: Cryptographic hash functions do not have any collisions.
- (k) TRUE or FALSE: User chosen passwords are good seed values for a PRG.
- (l) TRUE or FALSE: A website that rejects all user input that contains `<script>` tags may still be vulnerable to XSS attacks.
- (m) TRUE or FALSE: DNSSEC prevents a man-in-the-middle from seeing what names someone is looking up.

- (n) TRUE or FALSE: W^X (non-executable memory) is a good defense against return-oriented programming.
- (o) TRUE or FALSE: In DNSSEC, NSEC3 records effectively prevent zone enumeration.
- (p) TRUE or FALSE: A NIDS is preferable to a HIDS for HTTPS traffic.
- (q) TRUE or FALSE: The same-origin policy does not prevent <https://abc.com> from loading an image from <http://xyz.com> onto its page.
- (r) TRUE or FALSE: In the event where the domain name to IP address binding has changed, the DNS server must send invalidation messages to clients in order to clear their DNS cache.
- (s) TRUE or FALSE: ASLR randomizes the location of program's stack in memory, and therefore prevents the attacker from correctly guessing the location of the return address to modify.
- (t) TRUE or FALSE: Setting the NX bit (i.e. disabling executable permission) on pages spanning the program's stack prevents all buffer overflow attacks.
- (u) TRUE or FALSE: Consider a program that is compiled with stack canary protection. The canary has the same value across multiple executions of the program.
- (v) TRUE or FALSE: Content Security Policy mitigates against cross-site scripting attacks by only allowing a whitelist of script sources.
- (w) TRUE or FALSE: Setting the httpOnly flag in a cookie prevents CSRF attacks by preventing the victim's cookies from being sent along with the request.

Problem 2 *Short questions*

(16 points)

- (a) A NetGear router runs a web server for local users to manage the device in the home. If an attacker gets a local user to visit a page that somewhere includes the URL `http://{RouterIP}/cgi-bin/;COMMAND` the router will execute this command. What type of attack is this?
- (b) The tools you would need to break federal wiretap law are (Circle one):
1. free and widely available
 2. only available to government agencies
 3. only available on the dark web
- (c) Which of these can encrypt messages of arbitrary size? Select zero or more.
1. a block cipher
 2. a block cipher with a mode of operation
 3. a stream cipher
 4. RSA
- (d) If the Facebook abuse protocol wants to enable the recipient of a message to prove to a third party that this was the message he/she received, should it use a MAC, a digital signature, or either of the two at the sender?

- (d) Instead you propose changing to use HMAC-DRBG using SHA256 which is well seeded. Your manager thinks this is a good idea, but that you should also update the pRNG with the current time after each message. Does this update weaken the security? Why or why not?
- (e) For hashing passwords of users, LoadedDice claimed to follow “best practices”, computing the hashed password as $\text{SHA-256}(\textit{password})$ for all users. This is obviously not true, so what two things are wrong with their hashing practices?
- (f) Unfortunately gambling is illegal, and the feds gained a wiretap order which they already used to record all of LoadedDice’s traffic. LoadedDice’s web site used RSA-AES-SHA256 as the TLS parameters. If the feds later execute a search warrant will they be able to decrypt traffic? Why or why not?

- (c) Your security posture doesn't just stop external threats but also internal damages. Human Resources reports that a worker has complained about another worker viewing inappropriate web pages at work. Which system can substantiate if this claim is correct?
- (d) One of your vendors offers an IPS (Intrusion Prevention System) solution which you consider installing. The vendor claims that it has a zero false negative rate when detecting attacks by the RedBand. and a false positive rate of less than .1%, however false positives are very disruptive and will cause a lot of collateral damage. The vendor is honest but will provide no other information before you decide to deploy it. Is this a good system? Why or why not?

Problem 5 *Tasty but insecure*

(16 points)

Oski was hired as a security consultant for `www.tastytreats.com`. The developer of this website has not taken CS 161 so none of the user input is validated or escaped. For each question, state a way that Oski can test for the specified vulnerability.

- (a) After talking to an engineer at Tasty Treats, Oski finds the following line of code running at the web server:

```
treat = "SELECT * FROM TREATS WHERE Name = ' " + treatName + "';";
```

What could Oski enter in the place of `treatName` to erase all treats from the database?

- (b) What should Oski recommend that Tasty Treats do to prevent the attack in part (a)? Write exactly two different defenses, and not more.

- (c) Oski wants to look up the treat, `samosas`, on the Tasty Treats website. When he enters `samosas` in the search bar he notices that his browser issues an http request to the url

```
http://www.tastytreats.com/search.html?term=samosas
```

and that the web server embeds the term with no validation in the page returned. For example, he gets the following message:

```
The treat "samosas" was not found.
```

What kind of vulnerability has Oski found? Be as specific as you can in its name (e.g., a web attack is certainly not acceptable :P).

- (d) Consider an attacker who wants to obtain the cookies of Alice for `tastytreats.com`. Write down the URL that Oski (pretending he is an attacker) can send to Alice in an email such that, when she clicks on that URL, Oski's site (`www.oski.com/getcookie`) ends up obtaining her cookies. If you don't know the exact name of a HTML or Javascript command, write a name with a suggestive meaning.

Problem 6 *DNSSEC*

(12 points)

While browsing memes late at night, you make a DNSSEC-enabled request for the A record of `oskibear2k16.berkeley.edu`.

To your dismay and disappointment, you receive the response: `NXDOMAIN`. The response also contains `NSEC` and `RRSIG` records.

- (a) What further DNS queries, if any, must you make to validate this response? Assume all caches are empty. Provide both the domain(s) to be queried and the necessary record type(s) for each domain. You don't need to use all the lines below:

Record Type	Domain

- (b) What parties must you trust in order to validate this response and confirm that `oskibear2k16.berkeley.edu` really doesn't exist? Circle all that apply and provide no explanation.
 1. your local network administrator
 2. the DNS server that gave you this response
 3. one or more certificate authorities
 4. the DNS root zone administrator

- (c) The devious students at a certain university to the south decide that this will not do. They have found a way to spoof responses to any of your DNS requests, and furthermore, have stolen the private ZSK of the `.edu` zone!

Now when you send a request for `oskibear2k16.berkeley.edu`, you get a response with an A record of `171.64.64.64` and an `RRSIG` that your resolver says is valid.

How did this happen? Be specific. (Once again, assume that all caches are empty.)

Problem 7 *To script or not to script?***(12 points)**

Our favorite social media platform, FacePalm, feels that having user-defined Javascript running on its page would enhance user experience. They're thinking of implementing a new feature where users can enter *script-mode* in which any posts containing script tags will have their scripts run on the page. So, for example, if a user posts `<script> alert(1); </script>`, a user viewing the post in script mode would see the alert, whereas a user not browsing in script mode would see sanitized output.

- (a) Their initial release of the feature has users enter script-mode by navigating to <https://facepalm.com/script-mode>, whereas normal browsing navigates to <https://facepalm.com/no-script-mode>. They quickly notice scripts popping up to steal the cookies of users who enter script-mode. How could they change the cookies they set for users such that attackers would not be able to steal cookies? Answer in one sentence.

- (b) After implementing this change, attack-scripts no longer steal cookies, but they quickly notice scripts popping up that send users' private information (i.e. birthday, apps visited, etc.) over to <https://evil.com>. The FacePalm admins decide that users browsing in script-mode should not be allowed to view their own private content. They decide to switch the URLs that users browse to: now to browse in script-mode, users visit <https://script-mode.facepalm.com>, whereas for normal browsing, users now visit <https://no-script-mode.facepalm.com>, and to view private content, users visit <https://no-script-mode.facepalm.com/private>. Why can attackers no longer steal private information?

- (c) After the change in part b, FacePalm notices scripts that force users to post annoying cypypasta. After seeing "I am still getting ya" posted for the fifteenth time, the FacePalm admins decide to put an end to it, so they restrict post-content requests to no-script-mode domains (i.e. users must request <https://no-script-mode.facepalm.com/post> in order to post content, and there is no script-mode equivalent endpoint). However, they notice that benign users still end up posting annoying messages to <https://no-script-mode.facepalm.com/post>. What type of vulnerability is FacePalm likely facing here, and how could they solve this issue?

Problem 8 *Diagnosing Heartbleed*

(19 points)

OpenSSL, a popular open-source implementation of TLS, was recently attacked by exploiting a vulnerability in its handling of heartbeat messages, which is now known as the Heartbleed vulnerability.

Someone decided that TLS should support its own “heartbeat” messages to maintain long term connections (in addition to TCP’s keepalives) and that these heartbeat messages should contain arbitrary content sent by the client and echoed back by the server.

The relevant, simplified implementation is as follows:

```
typedef struct {
    /* length of heartbeat message from client. This field is set by the server. */
    uint32_t length;
    /* contents of the heartbeat message. This is controlled by the malicious client. */
    uint8_t *data;
} SSL3_RECORD;

uint8_t *processHeartbeat(SSL3_RECORD *r)
{
    /* message type e.g. HEARTBEAT; this field set by client */
    uint8_t type = *((uint8_t *) &r->data[0]);
    /* length of data; this field set by client */
    uint16_t len = *((uint16_t *) &r->data[1]);
    uint8_t *buf = malloc(len + 3);
    /* type of response */
    *((uint8_t *) &buf[0]) = HEARTBEAT_RESPONSE;
    /* response length equals client's message length */
    *((uint16_t *) &buf[1]) = len;
    /* copy the client's message verbatim */
    memcpy(buf + 3, r->data, len);
    /* return the message to be sent to the client */
    return buf;
}
```

Figure 1: Vulnerable Procedure

Observe that the `SSL3_RECORD` contains a pointer to data sent by the malicious client. The server populates the length field of `SSL3_RECORD` with the length of the data actually sent by the client. The data sent by the client itself contains the encoded message itself, with the first byte always specifying the type of message. For Heartbeat messages the type specifies that it is a heartbeat message, the next two bytes specify the length the client expects back, and the remaining bytes are the nonce the client sends.

- (a) A vulnerability here makes the program leak large parts of its memory upon processing a heartbeat message. This is catastrophic as the memory may potentially have cryptographic keys, session cookies from other users, and other secrets. Describe the vulnerability and construct the exploit heartbeat message.

Problem 10 *Hunting block cipher chaining modes* (12 points)

Unexcited about the block cipher chaining modes Alice learned in CS 161, especially CBC and CTR, she decides to create her own that also provides IND-CPA security. Let B be a block cipher and denote by $B_k(p)$ the block cipher applied to plaintext p with key k . Alice has come up with the block cipher chaining modes below and she needs your help to decide if they are IND-CPA or not. For each scheme below, say whether it is IND-CPA or not. If it is not IND-CPA, additionally give a concrete example of an information the attacker learns about the plaintext from the ciphertext in this encryption scheme that he/she would not be able to learn from an IND-CPA-secure scheme.

Consider the plaintext $P = (P_1, P_2, \dots, P_\ell)$ where each P_i has size the block size of B . Let \oplus be bitwise XOR.

(a) $Enc_k(P) = (C_1, \dots, C_\ell)$, where $C_i = B_k(i \oplus P_i)$.

(b) $Enc_k(P) = (IV_1, C_1, \dots, IV_\ell, C_\ell)$, where $C_i = B_k(IV_i \oplus P_i)$, where each IV_i is generated randomly and independently of the other IVs.

(c) Let $IV = hash(P)$, where hash is a cryptographic hash function (in particular, it is collision resistant). Assume the hash of P fits exactly in the IV size. Let $C_0 = IV$. $Enc_k(P) = (IV, C_1, \dots, C_\ell)$, where $C_i = B_k(C_{i-1} \oplus P_i)$.

Problem 11 *Sessions with localStorage*

(15 points)

Browsers support a feature called `localStorage`, which allows a website to save and load data on the user's computer. The storage is a key-value database, which can be accessed by JavaScript code on a webpage. Data that is saved by one page can be accessed by any other page that comes from the same origin. Unlike cookies, data in `localStorage` is not automatically sent with any HTTP requests.

Consider building a website using `localStorage` to implement a session management system.

- (a) Suppose the server receives an HTTP request containing correct credentials that authenticate a user. The server generates a random session token to be stored in the browser. However, the server doesn't have direct access to `localStorage`.

Describe an HTTP response that the server can send in order to get the session token stored. You can either provide the code snippet or describe what the code should do.

- (b) Next, the user loads a page with a form for executing an action while logged in. The page always has the exact same URL and HTML code, regardless of which user is logged in (this helps with caching performance).

Describe what could be included in the page to ensure that a form submission to the site's server will include the user's session token, as well as how the server would receive the session token. You don't have to write actual code; assume that JavaScript code is capable of:

- Running when the page loads
- Running just before the browser submits a form
- Saving and loading data in `localStorage`
- Inspecting and modifying the contents of the webpage as it is interpreted by the browser (this is called the DOM or Document Object Model).

- (c) Consider how an XSS vulnerability would affect the security of this session management system, compared to a session management system that stores a session token in a cookie. Could an attacker exploit an XSS vulnerability to steal the session token in this system? Could an attacker steal the session token if the site stored the session token in a cookie instead?
- (d) Consider web-specific attacks, other than XSS, that an attacker might perform in order to have the site take action on behalf of the user without the user's knowledge. Name one type of attack in that is a problem with using cookies that using Local-Storage to record session IDs prevents.
- (e) Name one type of attack other than XSS that is equally applicable to this system as it is to a system that uses cookies.