Popa & Weaver
Spring 2019

CS 161
Computer Security

Final Exam

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

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

SIGN your name: ________________________________

PRINT your SID: ________________________________

Name of the person sitting to your left: ______________________ Name of the person sitting to your right: ______________________

You may consult **three** double-sided, handwritten sheet of paper of notes. You may not consult other notes or textbooks. Calculators, computers, and other electronic devices are not permitted.

Bubble every item completely. Avoid using checkmarks or Xs.
If you want to unselect an option, erase it completely and clearly.

For questions with **circular bubbles**, you may select only one choice.

○ Unselected option (completely unfilled)

● Only one selected option (completely filled)

For questions with **square checkboxes**, you may select one or more choices.

■ You can select

■ multiple squares (completely filled).

If you think a question is ambiguous, please come up to the front of the exam room to the TAs. If we agree that the question is ambiguous we will announce the clarification to everyone.

You have 170 minutes. There are 10 questions of varying credit (125 points total).

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

Problem 1 ***Potpourri is unhealthy*** **(20 points)**

(a) (2 points) True or False: Modern web browsers protect against clickjacking by using the same-origin policy to prevent sites from putting other origins into an `iframe`.

○ True ○ False

(b) (2 points) True or False: The primary danger of XSS vulnerabilities is that they let an attacker execute Javascript on the victim machine without having the victim visit the attacker's website.

○ True ○ False

(c) (2 points) True or False: Even if you carefully inspect all links that you click, you can still be vulnerable to a CSRF attack.

○ True ○ False

(d) (2 points) True or False: For most common implementations of session cookies (as seen in lecture and on the project), a SQL injection can let an attacker steal the sessions of other users.

○ True ○ False

(e) (2 points) True or False: If a script is loaded from another origin using a `script` tag, the same-origin policy prevents this script from reading the cookies on the current page.

○ True ○ False

(f) (2 points) Some architectures prohibit executing unaligned machine code instructions. This makes it harder for an attacker to perform ___(1)___, which often chains together "gadgets" found by jumping to the middle of instructions.

(1):

(g) (2 points) In certificate transparency, after a certificate authority signs a certificate, they submit the signed certificate to a certificate transparency log. They receive a(n) ___(1)___ in return. If the signed certificate is not in the log after a certain amount of time, certificate authorities can use this to prove malicious or incorrect behavior of the log.

(1):

(h) (2 points) The use of trusted boot systems and signed code helps prevent ___(1)___, which is malcode that often hides in the BIOS and operating system.

(1):

(i) (2 points) At the beginning of their life cycles, computer worms grow ___(1)___, but as time goes on it becomes harder to find new victims and the worm growth slows.

(1):

(j) (2 points) Tor is fundamentally vulnerable against timing attacks conducted by global adversaries because it is supposed to be ___(1)___.

(1):

Problem 2 ***Welcome to the Wonderful World of*** **(14 points)**

People of earth, boys and girls, children of all ages, welcome to the wonderful world of block cipher, symmetric encryption, and hash functions!

(a) There are two **symmetric** encryption schemes, SymEncA and SymEncB. Both implement valid encryption / decryption on a message / ciphertext, but one of them may be insecure.

Bob wants to combine these two schemes to avoid the risk of using a failed encryption scheme. He proposes the following combinational construction.

Construction I: The ciphertext of the message M consists of two parts:

1. The first part of the ciphertext C_{part-1} = SymEncA.Encrypt($k; M$).
2. The second part of the ciphertext C_{part-2} = SymEncB.Encrypt($k; M$).
3. That is, the ciphertext is $C = (C_{part-1}, C_{part-2})$.

⋄ **Question:** Is the Construction I secure if at least one of the symmetric encryption schemes is secure? Why?

- If yes, fill the corresponding circle, and provide a concise description of why it can hide the message.
- If no, fill the corresponding circle, and provide a concise description of a counterexample. Please answer within 4 lines.

○ Yes. ○ No.

Please answer within the following four lines.

____

____

____

____

(b) Bob proposes another combinational construction.

Construction II: To encrypt message M, there are two steps:

1. The intermediate value I = $\mathsf{SymEncA.Encrypt}(k; M)$, which means it encrypts M directly under key k. This intermediate value is not the ciphertext.
2. The final ciphertext C = $\mathsf{SymEncB.Encrypt}(k; I)$, which means it encrypts the intermediate value under key k.
3. That is, the ciphertext is C = $\mathsf{SymEncB.Encrypt}(k; \mathsf{SymEncA.Encrypt}(k; M))$

⋄ **Question:** Is the Construction II secure if at least one of the symmetric encryption schemes is secure? Why?

- If yes, fill the corresponding circle, and provide a concise description of why it can hide the message.
- If no, fill the corresponding circle, and provide a concise description of a counterexample.

○ Yes. ○ No.

Please answer within 4 lines.

____

____

____

____

(c) You accidentally fell into a trap and entered the 8th floor of Soda Hall.

On the wall the following sentences appear:

No block cipher provides IND-CPA confidentiality because they must be deterministic.

No hash function provides IND-CPA confidentiality because they must be deterministic.

No HMAC provides IND-CPA confidentiality because ...

No digital signature provides IND-CPA confidentiality because ....

Some words on the last two lines are missing.

⋄ **Question:** What do you think the reasons should be? (Answer within the lines)

- *No HMAC provides IND-CPA confidentiality because:*

  ____

- *No digital signature provides IND-CPA confidentiality because:*

  ____

  ____

(d) To make RSA signatures secure, we can apply a cryptographic hash function H over the message M, where the output of the hash function is a non-negative integer in $\{0, 1, ..., 2^{256} - 1\}$. We know that this hash function H must be second-preimage resistant; otherwise, another message $M' \neq M$ can be found that also matches the signature.

Later, the RSA signature is computed as follows:

$$sig = H(M)^d \pmod{n},$$

where n is the RSA modulo, (e, n) forms the RSA public key, (d, n) forms the RSA private key, and $ed \equiv 1 \pmod{\phi(n)}$, where $\phi(n)$ is Euler's totient function.

Alice wonders whether she can customize her hash function. She creates another function H', modified from H:

$$H'(x) = H(x) - H(''\texttt{Alice}'') \pmod{2^{256}}.$$

⋄ **Question:** Can we use H' instead of H for RSA signature for *every* possible message that Alice might sign?

○ Yes. ○ No.

And explain within the line:

____

Problem 3 ***Low-level Denial of Service*** **(8 points)**

In this question, you will help Mallory develop new ways to conduct denial-of-service (DoS) attacks.

(a) CHARGEN and ECHO are services provided by some UNIX servers. For every UDP packet arriving at port 19, CHARGEN sends back a packet with 0 to 512 random characters. For every UDP packet arriving at port 7, ECHO sends back a packet with the same content.

Mallory wants to perform a DoS attack on two servers. One with IP address A supports CHARGEN, and another with IP address B supports ECHO. Mallory can spoof IP addresses.

i. Is it possible to create a single UDP packet with no content which will cause both servers to consume a large amount of bandwidth?

- If yes, mark 'Possible' and fill in the fields below to create this packet.
- If no, mark 'Impossible' and explain within the provided lines.

○ Possible ○ Impossible

If possible, fill in the fields:

Source IP: ______________ Destination IP: ______________
Source port: ______________ Destination port: ______________

If impossible, why?

____

____

ii. Assume now that CHARGEN and ECHO are now modified to only respond to TCP packets (post-handshake) and not UDP. Is it possible to create a single TCP SYN packet with no content which will cause both servers to consume a large amount of bandwidth?

- If yes, mark 'Possible' and fill in the fields below to create this packet.
- If no, mark 'Impossible' and explain within the provided lines.

○ Possible ○ Impossible

If possible, fill in the fields:

Source IP: ______________ Destination IP: ______________
Source port: ______________ Destination port: ______________
Sequence #: ______________ Ack #: N/A

If impossible, why?

____

____

(b) A typical web server maintains a connection after receiving each TCP connection request. Write down the the name of the transport layer attack that can cause denial-of-service on the web server which works by consuming a large amount of server memory.

____

Problem 4 ***OTP-KE*** **(9 points)**

Alice and Bob want to communicate securely. They come up with a new key exchange protocol, inspired by the Diffie-Hellman key exchange but based on the security properties of the one-time pad. Assume $E_K(M)$ is a one-time-pad with message M and key K. The two of them randomly generate A and B, which will be their own unique one-time pad keys. Alice also generates a truly random key S, which is the symmetric key she and Bob want to agree on and will be used for further communication after the key exchange.

To execute the protocol, Alice uses one-time-pad encryption to encrypt S using her secret key A, then sends $E_A(S)$ to Bob. Bob encrypts the resulting message using his secret key and sends back $E_B(E_A(S))$. Alice decrypts that message and sends back $D_A(E_B(E_A(S)))$.

Please answer each of the following questions in three sentences or less. Longer responses will not get credit.

(a) Explain how Alice and Bob can agree on S based on this protocol.

(b) Is this protocol secure against a passive attacker?

○ Yes ○ No

If yes, explain why. If no, provide an attack.

(c) Is this protocol secure against an active attacker?

○ Yes ○ No

No explanation needed.

Problem 5 ***Private set intersection*** **(13 points)**

Suppose Alice has a list of n integers $a_1, a_2, \ldots, a_n$; and Bob has a list of n integers as well $b_1, b_2, \ldots, b_n$. Each integer is only 16 bits long.

(a) Alice wants to know if they have any numbers in common, i.e., if there exist i, j such that $a_i = b_j$. Bob applies a function F to each of his numbers, and sends the list $F(b_1), F(b_2), \ldots F(b_n)$ to Alice.

i. Which of the following choices of F allows Alice to identify whether Bob has a b_j that is equal to some element a_i in Alice's list? k is a shared symmetric key.

☐ $F(x) = \text{SHA-256}(x)$

☐ $F(x) = \text{AES-CBC}_k(x)$

☐ $F(x) = \text{SHA-256}(x\|r)$, where r is 256 bits long and randomly chosen per x

☐ $F(x) = \text{SHA-256}(x\|k)$

☐ $F(x) = \text{AES}_k(x)$

☐ None of the above

ii. Which of the following choices of F ensure that Alice can **only** identify the b_j values that are equal to some element a_i in Alice's list? Alice should **not** be able to identify the value of b_j if it is not equal to some value in her list.

☐ $F(x) = \text{SHA-256}(x)$

☐ $F(x) = \text{AES-CBC}_k(x)$

☐ $F(x) = \text{SHA-256}(x\|r)$, where r is 256 bits long and randomly chosen per x

☐ $F(x) = \text{SHA-256}(x\|k)$

☐ $F(x) = \text{AES}_k(x)$

☐ None of the above

(b) Now suppose that Alice and Bob **both** wish to learn the common elements in their lists. To this end, they engage in a new protocol inspired by Diffie/Hellman. They agree on a large prime number p. Alice chooses a secret value α uniformly at random from the set $\{1, 2, 3, \dots, p-2, p-1\}$. Bob follows the same procedure to choose a secret value β. They then exchange four messages sequentially, as follows. (H is a secure hash function.)

1. Alice ⟶ Bob: $(H(a_1))^\alpha, (H(a_2))^\alpha, \dots, (H(a_n))^\alpha$ (all modulo p)
2. Bob ⟶ Alice: $(H(b_1))^\beta, (H(b_2))^\beta, \dots, (H(b_n))^\beta$ (all modulo p)
3. Alice ⟶ Bob: ????????????????????????????????????
4. Bob ⟶ Alice: ????????????????????????????????????

i. What values should Alice and Bob send to each other in steps 3 and 4? They should be able to identify values that exist in both their lists. They should **not** be able to identify any value in the other person's list if is not equal to some value in their own list.

3. Alice ⟶ Bob: ____

4. Bob ⟶ Alice: ____

ii. Now suppose that Bob decides to cheat in step 4. Instead of sending the correct message to Alice, he wishes to make Alice believe that their lists are identical. Alice follows the protocol as before, and does not expect Bob to cheat.

◇ **Question:** What values should Bob send to Alice in step 4 to achieve this?

____

Problem 6 ***Network Security*** **(20 points)**

Answer the following questions about network security.

(a) Bob connects his laptop to the DeCafe coffee shop's Wifi, which anyone nearby can join without a password. He browses to the website `http://www.foocorp.com`. At the table next to him is an evil attacker, Mallory, who has also joined the DeCafe Wifi network. What kind of threat model best describes Mallory when she first joins the network, with respect to Bob's connection with DeCafe router's?

- ○ Off-path attacker
- ○ In-path attacker
- ○ On-path attacker
- ○ None of these

(b) Bob returns home and types into his browser `www.foocorp.com`. Suppose that Mallory has managed to poison the DNS cache on Bob's laptop, such that it now thinks the IP address of `www.foocorp.com` is 6.6.6.6, which is the IP address of a server that Mallory controls.

- ☐ Mallory will be unable to steal Bob's cookies for `http://www.foocorp.com` if `http://www.foocorp.com` uses HTTP-Only cookies.
- ☐ Mallory will be unable to steal Bob's cookies for `http://www.foocorp.com` if `http://www.foocorp.com` uses a CSP policy that only allows scripts to be loaded from sources on `foocorp.com`
- ☐ Mallory will be unable to steal `foocorp.com` cookies marked with the `secure` flag.
- ☐ Mallory will be unable to inject JavaScript into `http://www.foocorp.com`
- ☐ Mallory will be unable to steal Bob's `foocorp.com` cookies if `foocorp.com` uses HTTPS and Bob's browser checks certificate transparency logs over HTTPS.
- ☐ Mallory will be unable to steal Bob's cookies if `foocorp.com` uses HTTPS and Bob's browser has previously received an HSTS header.
- ☐ None of the above

(c) Suppose that `foocorp.com` domain has the following four subdomains:
(`www`, `alphabet`, `sushi`, `money`).
The attacker knows that `foocorp.com` has only four subdomains but does not know any of their names, and wishes to discover the subdomains using the zone enumeration attack discussed in class.

⋄ **Question:** Assuming every DNS server uses plain NSEC, what is the minimum number of queries the attacker needs to make to `foocorp.com`'s nameservers in the worst-case for the attacker?

- ○ 0
- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ○ 5
- ○ 6 to 10
- ○ 11 to 24
- ○ 25 to 35
- ○ ≥ 36

(d) Suppose that a user Alice is browsing the Internet at home and Mallory is an on-path attacker. In which of the following scenarios will Mallory be able to identify whether or not Alice is visiting a website on `foocorp.com`?

- ☐ Alice's machine and local DNS resolver randomize the source port of DNS queries; `foocorp.com`'s NS server use DNS (without DNSSEC); `foocorp.com` does not use HTTPS
- ☐ Alice's machine and local DNS resolver use a fixed source port for every DNS query; `foocorp.com`'s NS server uses DNSSEC with plain NSEC; `foocorp.com` does not use HTTPS
- ☐ Alice's machine and local DNS resolver use a fixed source port for every DNS query; `foocorp.com`'s NS server uses DNSSEC with NSEC3; `foocorp.com` does not use HTTPS
- ☐ Alice's machine and local DNS resolver use a fixed source port for every DNS query; `foocorp.com`'s NS server uses DNSSEC with NSEC3; `foocorp.com` uses HTTPS
- ☐ None of the above

(e) `FooCorp` has chosen to use very short TTLs in all of their DNS responses. Which of the following statements are true?

- ☐ Short TTLs help protect against attacks where `FooCorp`'s DNS servers have been compromised
- ☐ Assuming all DNS servers used DNSSEC with plain NSEC, then `FooCorp`'s decision to use short TTLs will increase the amount of work that the DNS servers of `FooCorp`'s parent zone need to perform
- ☐ Short TTLs increase the number of requests `FooCorp`'s DNS servers need to support
- ☐ Short TTLs help protect against DNS cache poisoning attacks by an on-path attacker
- ☐ Short TTLs help protect against blind-spoofing attacks
- ☐ None of the above

(f) `FooCorp` hosts *all* of its servers on machines provided by CheapCloud: a large, but unreliable, cloud hosting provider. CheapCloud suffers from two major problems: (i) they have frequent data breaches; and (ii) they often need to assign new IP addresses to their customers' servers. Nevertheless, CheapCloud promptly notifies their customers whenever either of these events occurs.

⋄ **Question:** Which of the following designs or techniques can `FooCorp` use to help mitigate some of the security issues caused specifically by CheapCloud's poor environment?

- ☐ `FooCorp` uses plain DNS and sets short TTLs for all of its DNS responses
- ☐ `FooCorp` uses RSA-based TLS with certificate pinning
- ☐ `FooCorp` uses DNSSEC with plain NSEC
- ☐ `FooCorp` uses DHE-based TLS, but does not use certificate pinning
- ☐ `FooCorp` uses DNSSEC with NSEC3
- ☐ None of the above

(g) Suppose `foocorp.com`, `.com`, and the root DNS servers all use DNSSEC. An attacker has compromised the `.com` zone's DNS servers and stolen just the `.com` Zone Signing Key (ZSK). Once `.com` manages to remove the attacker, which of the following steps should be taken to prevent an attacker from using the stolen ZSK to forge DNS responses after all existing signatures have expired?

- ☐ `foocorp.com` will need to update its RRSIG records
- ☐ `foocorp.com` will need to update its DNSKEY records
- ☐ `foocorp.com` will need to update its DS records
- ☐ `foocorp.com` will need to update its Key Signing Key
- ☐ `.com` will need to update its DNSKEY records
- ☐ `.com` will need to update its RRSIG records
- ☐ `` `.' `` (the root zone) will need to update its DNSKEY records
- ☐ `` `.' `` (the root zone) will need to update its DS records for `.com`
- ☐ None of the above

Problem 7 ***Detection to Surveillance*** **(7 points)**

The "No Such Agency" is looking to build a new surveillance system designed to detect "bad dudes". They want to deploy this system at a single location on the network that they identified as a hub for international communication.

(a) One proposed detector has a false positive rate (FPR) of X, and a false negative rate (FNR) of Y, and the other proposed detector has a FPR of Y and a FNR of X. Let C_P be the cost of a false-positive, C_N be the cost of a false negative, and p be the fraction of malicious communications. Assume the detectors are otherwise identical.

◇ **Question:** For what value of p are the two systems equally preferred (as a function of X, Y, C_P and C_N)?

p = [ ]

Ungraded scratch space for calculations:

(b) Someone else suggests alerting at random: a random system will alert with probability r, and will not alert with probability $(1 - r)$. Find the false-positive and the false-negative rates of this system.

FPR = [ ]

FNR = [ ]

Ungraded scratch space for calculations:

Problem 8 *Virtual Tables, Real Fun* (16 points)

The following code runs on a 32-bit x86 system.

```c
#include <stdio.h>
int main() {
    FILE *fp;
    char buf[8];
    fp = fopen("outis", "rb");
    fread(buf, sizeof char, 12, fp);
    fclose(fp);
}
```

Behind the hood, the `FILE` struct is implemented in `stdio.h` as follows:

```c
struct _IO_FILE; /* implementation omitted */

typedef struct {
    struct _IO_FILE ufile;
    struct _IO_jump_t *vtable;
} FILE;

struct _IO_jump_t {
    size_t (*fread)(void *, size_t, size_t, FILE *);
    size_t (*fwrite)(void *, size_t, size_t, FILE *);
    int (*fclose)(FILE *);
    /* more members below omitted */
};

int fclose(FILE *fp) { return fp->vtable->fclose(fp); }
/* more implementations below omitted */
```

Make the following assumptions:

1. No memory safety defenses are enabled.
2. The compiler does not perform any optimizations, reorder any variables, nor add any padding in between `struct` members.
3. The implementation of the function `fopen` has been omitted. Assume a sensible implementation of `fopen` that initializes the `ufile` and `vtable` fields of the `FILE` struct to sensible values.

(a) Running the program in `gdb` using `invoke -d` as in Project 1, you find the following:

- `&buf = 0xbf608040`
- `&fp = 0xbf608048`
- `sizeof(struct _IO_FILE) = 32`

You wish to prove you can exploit the program by having it jump to the memory address `0xdeadbeef`. Complete the Python script below so that its output would successfully exploit the program.

NOTE: The syntax `\xRS` indicates a byte with hex value `0xRS`.

```
#!/usr/bin/env python2
import sys
sys.stdout.write('\x_____\x_____\x_____\x_____' + \

                 '\x_____\x_____\x_____\x_____' + \

                 '\x_____\x_____\x_____\x_____')
```

(b) Now you wish to write an exploit script, such that running it will successfully exploit the program. You save your code from part (a) as a script called `egg`. The vulnerable program is called `hack_me`. Which of the following code snippets is a valid exploit script?

☐
```
#!/bin/bash
./egg | invoke hack_me
```

☐
```
#!/bin/bash
invoke -e outis=$(./egg) hack_me
```

☐
```
#!/bin/bash
outis=$(./egg)
invoke hack_me $outis
```

☐
```
#!/bin/bash
./egg > outis
invoke hack_me
```

(c) Which of the following defenses would stop your attack in part (a) from exploiting the program by jumping to memory address `0xdeadbeef`? Assume `0xdeadbeef` is at a read-only part of memory.

☐ Stack canaries

☐ W^X

☐ ASLR which does not randomize the `.text` segment (as in Project 1)

☐ ASLR which also randomizes the `.text` segment

(d) (Consider this question independently from part (c).) Now consider that we move the variables `fp` and `buf` outside of the `main` function, as follows:

```c
#include <stdio.h>
char buf[8]; /* &buf = 0x08402020 */
FILE *fp;    /* &fp  = 0x08402028 */
int main() { /* rest of main is the same, but no variables */ }
```

TRUE or FALSE: It is possible to modify the exploit in part (a) to exploit this modified program.

○ TRUE

○ FALSE

Problem 9 ***Hacking the 161 Staff*** **(10 points)**

After months of development, the CS 161 staff is ready to unveil their new course homepage at http://cs161.org. Each TA has their own account and, after authenticating on http://cs161.org/login, can update any student's grade on the final exam by making an HTTP GET request to:

http://cs161.org/updatefinal?sid=<SID>&score=<SCORE>

where <SID> is the student ID, and <SCORE> is the student's new exam score (as a number – without the percent sign).

(a) Mallory is a student in CS 161, with the student ID of 12345678. She wants to use a CSRF attack to change her exam score to 100 percent. She overhears her TA mention in discussion that he likes to visit http://cool-web-forum.com which Mallory happens to know does not properly sanitize HTML in user inputs.

◇ **Question:** Give an input which Mallory can post to the forum in order to execute a CSRF attack to change her exam score, assuming there are no CSRF defenses on cs161.org.

____

____

(b) The TA then visits the web forum, yet Mallory's grade does not change. Mallory deduces that the 161 staff must have included a defense for CSRF on their webpage. Not one to be deterred, Mallory decides to attempt her attack again.

The login page has an *open redirect*: It can be provided a webpage to automatically redirect to after the user successfully authenticates. For example the URL:

http://cs161.org/login?to=http://google.com

would redirect any logged in user to http://google.com.

Using this information, Mallory crafts the following attack—replacing your URL in part (a) with the following URL:

http://cs161.org/login?to=http://cs161.org/updatefinal?sid=12345678&score=100

A few minutes later, Mallory observes that her final grade is changed to a 100 percent. Which of the following are CSRF defenses that Mallory might have circumvented?

☐ Origin checking ☐ Content-Security-Policy ☐ Cookie policy

☐ Referer checking ☐ Prepared statements ☐ Same-origin policy

☐ CSRF tokens ☐ Session cookies ☐ None of the above

(c) The 161 staff update their site to better protect against CSRF. Mallory now notices that the website contains a profile page for each member of the 161 staff, reachable from the URL

`http://cs161.org/staff?name=<name>`

where `<name>` is replaced with each staff member's name. If the provided `<name>` does not correspond to a member of the 161 staff, then instead a page is loaded with a message stating "Sorry, but there is no TA named `<name>`!"

Suspecting that this website might be vulnerable to reflected XSS, Mallory visits the following URL:

`http://cs161.org/staff?name=<script>alert(0);</script>`

A Javascript popup immediately appears on her screen. Mallory smiles, realizing that she can weaponize this to login as her TA. She returns to the web forum that her TA frequently visits and posts a link.

Assume that Mallory's TA will click on any link that he sees on the web forum, and assume that Mallory controls her own website `http://mallory.com`.

⋄ **Question:** How can Mallory pull off her attack and login as her TA? Make sure to include the link she posts on the forum in your answer. If you assume that Mallory's website has any scripts running, you must define what they are and what inputs they take in.

Problem 10 ***Evil TLS*** **(8 points)**

(a) A company wants to protect their web server by installing a new NIDS that will man-in-the-middle and decrypt all HTTPS traffic sent to its web server. The connections are end-to-end encrypted between the clients and the web server, and the NIDS is installed at a location that can see all the encrypted traffic. The NIDS could be passive (only inspects traffic), or it could be active (dropping or injecting packets). If the company gives the NIDS access to the TLS private key for the server, the NIDS will be able to decrypt a TLS connection to the web server if the connection uses…

☐ RSA TLS, and the NIDS is passive.

☐ RSA TLS, and the NIDS is active.

☐ Ephemeral Diffie-Hellman TLS, and the NIDS is passive.

☐ Ephemeral Diffie-Hellman TLS, and the NIDS is active.

(b) Imagine that we modify the TLS handshake as follows. Now, the server will be the first to send its nonce R_s. Then, the browser will send both its nonce R_b and the encryption $\{PS\}_{K_{server}}$ of a fresh random PS value to the server. Finally, browser and server compute $R_s \oplus R_b \oplus PS$ and use this as the only input to the PRNG. The cipher and integrity keys for the connection will depend only on $PRNG(R_s \oplus R_b \oplus PS)$.

True or False: This modified handshake is vulnerable to a replay attack.

○ True

○ False

If yes, fill in the messages that would be sent when performing a replay attack. If not, explain why the scheme is still secure.

⋄ **If yes, fill in the messages**:

1. Server sends nonce: R_{s1}
2. Browser sends nonce: R_{b1}
3. Browser sends encrypted pre-master secret: $E_1 = \{PS_1\}_{K_{server}}$
4. ...
5. Server sends nonce: ________________________________
6. Browser sends nonce: ________________________________
7. Browser sends encrypted pre-master secret: ________________________________
8. ...

⋄ **If no, explain on these lines (concisely)**:

____

____

____

Selected C Manual Pages

```
FILE *fopen(const char *pathname, const char *mode);

    The fopen() function opens the file whose name is the string pointed
    to by _pathname_ and associates a stream with it.  If _mode_ is "rb",
    this opens the text file for reading.  The stream is positioned at the
    beginning of the file.

size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);

    The function fread() reads _nmemb_ items of data, each _size_ bytes long,
    from the stream pointed to by _stream_, storing them at the location
    given by _ptr_.

size_t fwrite(void *ptr, size_t size, size_t nmemb, FILE *stream);

    The function fwrite() writes _nmemb_ items of data, each _size_ bytes
    long, to the stream pointed to by _stream_, obtaining them from the
    location given by _ptr_.

int fclose(FILE *stream);

    The fclose() function flushes the stream pointed to by _stream_ and
    closes the underlying file descriptor.
```

.

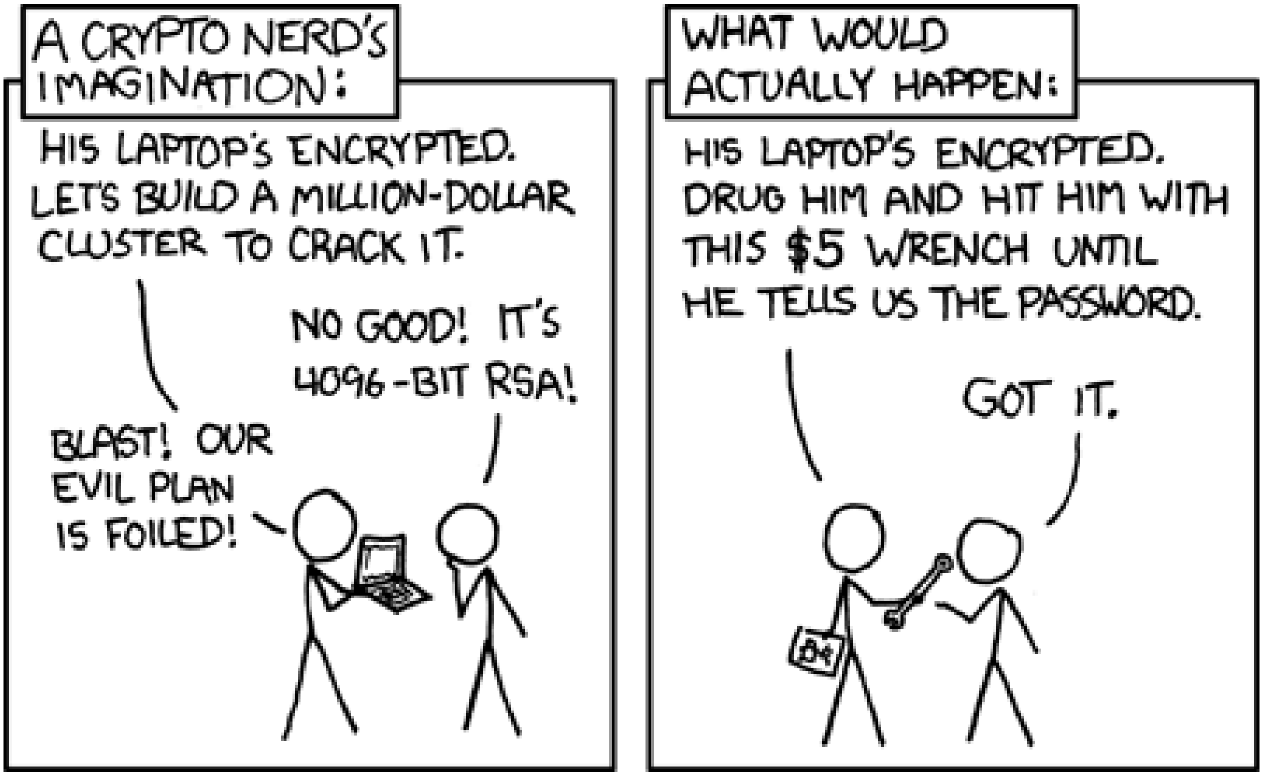

Figure 1: Actual actual reality: nobody cares about his secrets.
(Also, I would be hard-pressed to find that wrench for $5.)
(Also, why would anyone use a public key algorithm for disk encryption?)