Weaver Fall 2017 — 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 any academic misconduct on this exam will be reported to the Center for Student Conduct and may lead to a "F"-grade for the course. I am aware that Nick believes *in retribution.*

SIGN your name: _____

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

Your TA's name: _____

Number of exam of person to your left: _____

Number of exam of person to your right: _____

You may consult four sheets of notes (each double-sided). You may not consult other notes, textbooks, etc. Calculators, computers, and other electronic devices are not permitted. Please write your answers in the spaces provided in the test.

You have 180 minutes. There are 9 questions, of varying credit (248 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	6	7	8	9	Total
Points:	40	29	28	30	24	24	18	40	15	248
Score:										

Problem 1 ***True/False*** **(40 points)**

For each of the following, FILL IN THE BUBBLE next to **True** if the statement is correct, or next to **False** if it is not. Each correct answer is worth 4 points. Incorrect answers are worth 0 points. Answers left blank are worth 1 point.

(a) Stack canaries reliably block all stack overflow attacks.

○ **True** ● **False**

(b) Nonexecutable stacks reliably block all stack overflow attacks.

○ **True** ● **False**

(c) Nick's house is Ravenclaw ("Where those of wit and learning, will always find their kind").

○ **True** ● **False**

> **Solution:** Have you SEEN the rug in my office? Slytherin forever!

(d) Forward secrecy means that, if your private key is compromised, an attacker can't recover the plaintext for previous messages when they have also captured the ciphertext.

● **True** ○ **False**

(e) If you use a U2F security key as your second factor in a 2-factor supporting site, this generally prevents phishing attacks.

● **True** ○ **False**

(f) If a site uses DHE for TLS, an adversary who steals the site's private key can passively decrypt intercepted communications.

○ **True** ● **False**

(g) The Chinese "Great Firewall" operates using the same basic mechanism as a corporate firewall.

○ **True** ● **False**

> **Solution:** No. the Great Firewall is an on-path device that looks at requsets and injects replies, while corporate firewalls are in-path devices.

(h) If you combine two independent detectors in a way which reduces the false-positive rate this combination will increase the false-negative rate.

● **True** ○ **False**

(i) Using sprintf to format user input to an SQL query is safe if you just replace all ' characters with

’ in the user input.

○ **True** ● **False**

> **Solution:** Nope. A classic example is instead of the name being `robert';drop table students`, the name is `robert
> ';drop table students`

(j) Most of the block cipher modes you learned about require keeping the IV secret

○ **True** ● **False**

(k) When configuring a firewall, it's safer to use an approach based on blacklisting hosts than whitelisting hosts

○ **True** ● **False**

(l) HTTPS can prevent CSRF attacks

○ **True** ● **False**

(m) A secure hash function will only produce collisions with an infinitesmaly small probability.

● **True** ○ **False**

(n) AES-CTR mode provides integrity when properly used

○ **True** ● **False**

(o) AES-GCM mode provides integrity when properly used

● **True** ○ **False**

(p) PBKDF2 turns a password into a large amount of key material by repeatedly encrypting the password with AES.

○ **True** ● **False**

(q) If *Website A* loads a website from another domain (*Website B*) inside of an iframe, the same origin policy prevents Javascript from *Website B* from accessing any of the other website's content.

● **True** ○ **False**

(r) Consider a scanning worm that picks addresses uniformly at random. IPv6 changes IP addresses from 32b to 128b. Selecting an IPv6 address uniformly at random is not an effective strategy for a worm.

● **True** ○ **False**

Problem 2 ***Multiple Choice*** **(29 points)**

For multiple choice questions, select **all** which are correct.

(a) (1 point) (This question is *magic*, the amount of points may vary) The Magic Words are...

- ○ Livid Peregrine
- ● Squeamish Ossifrage
- ○ Senatorial Chickenhawk
- ○ Irate Vulture
- ○ None of the Above

(b) (4 points) Valid analogies between the Influenza virus and computer viruses include:

- ● Both are often detected based on "known bad" features on the virus
- ● Both may have designs which can mutate to evade detection
- ● Both spread faster when the percentage of vulnerable hosts is greater
- ○ Neither have caused substantial loss of life
- ○ None of the Above

(c) (4 points) DNSSEC, when validated only by the client, provides which of the following security properties for DNS responses? **Mark ALL that apply.**

- ○ Confidentiality
- ● Integrity
- ● Authentication
- ○ Availability
- ○ None of the Above

(d) (4 points) DNSSEC, when validated only by the recursive resolver, provides which of the following security properties for DNS responses? **Mark ALL that apply.**

- ○ Confidentiality
- ○ Integrity
- ○ Authentication
- ○ Availability
- ● None of the Above

(e) (4 points) Which of the following are IND-CPA when properly used?

- ○ ROT-13
- ● CBC
- ○ ECB
- ● CTR
- ● One Time Pad
- ● GCM
- ○ None of the Above

(f) (4 points) Which of the following are IND-CPA when the IV is mistakenly resused?

- ○ CBC
- ○ ECB
- ○ CTR
- ○ GCM
- ● None of the Above

(g) (4 points) Which of the following can defend against many heap overflow attacks

- ○ Stack Canaries
- ● ASLR
- ○ XSS
- ● Non executable heaps
- ● Memory safe languages
- ○ C++17
- ○ None of the Above

(h) (4 points) Which of the following attacks might allow an attacker to steal one of your browser's secure (HTTPS-only) cookies (**Mark ALL that apply**):

- ● Stored XSS
- ○ Clickjacking
- ● Reflected XSS
- ● Buffer overflow in your browser
- ○ Packet injection without exploitation
- ● Packet injection with a browser exploit
- ○ None of the Above

Problem 3 ***Moogle*** **(28 points)**

For the following questions, select **all** answers which apply.

(a) (4 points) Moogle is a search engine that claims to better than Google. Whenever a user searches for something, a line of text appears that says "You searched for: " followed by the user's query unescaped. Which of the following is this vulnerable to?

- ○ Command Injection
- ○ Buffer Overflow
- ○ Format String
- ○ CSRF
- ● XSS
- ○ SQL Injection
- ○ Clickjacking
- ○ None of the Above

(b) (4 points) Moogle uses the Maboody Rank Algorithm to order web pages. A part of this algorithm logs queries into a file. The code is shown below:

```
char* search(char* query) {
    char logging[100];
    sprintf (logging, "echo %s > log.txt", query);
    system(logging);
    return logging;
}
```

Which of the following is this vulnerable to?

- ● Command Injection
- ● Buffer Overflow
- ○ Format String
- ○ CSRF
- ○ XSS
- ○ SQL Injection
- ○ Clickjacking
- ○ None of the Above

(c) (4 points) Moogle decides to enable ASLR, NX, and stack canaries. Which of the following does this completely defend against?

- ○ Command Injection
- ○ Buffer Overflow
- ○ Format String
- ○ CSRF
- ○ XSS
- ○ SQL Injection
- ○ Clickjacking
- ● None of the Above

(d) (4 points) Moogle decides to use only Java as a programming language, they don't use the foreign function interface, and their Java compiler and runtime is bug free. Which of the following vulnerability does this completely defend against?

- ○ Command Injection
- ○ XSS
- ● Buffer Overflow
- ○ SQL Injection
- ○ Format String
- ○ Clickjacking
- ○ CSRF
- ○ None of the Above

(e) (4 points) Moogle decides to prevent other websites from including iframes to their page. Which of the following vulnerability does this completely defend against?

- ○ Command Injection
- ○ XSS
- ○ Buffer Overflow
- ○ SQL Injection
- ○ Format String
- ● Clickjacking
- ○ CSRF
- ○ None of the Above

(f) (4 points) Moogle decides to define a content security policy. What vulnerability does this partially defend against?

- ○ Command Injection
- ● XSS
- ○ Buffer Overflow
- ○ SQL Injection
- ○ Format String
- ○ Clickjacking
- ○ CSRF
- ○ None of the Above

(g) (4 points) Moogle decides to use hidden tokens inside their cookies to be sent along with their API requests. What vulnerability does this partially defend against?

- ○ Command Injection
- ○ XSS
- ○ Buffer Overflow
- ○ SQL Injection
- ○ Format String
- ○ Clickjacking
- ○ CSRF
- ● None of the Above

Problem 4 ***Banana Messenger*** **(30 points)**

The computer-firm **Banana Inc** has developed a messaging system called *bMessage* running on the amazingly secure BananaPhone. bMessage is designed to provide end-to-end protection between users so that Banana Inc can't read the messages.

The initial version of bMessage has a central keyserver, similar to the one in the project but slightly different. Since a user can have multiple devices, each with their own 2048b random RSA public key, when **Bob** queries for **Alice**'s public key, Bob doesn't just get one key but gets a list of keys, all of which belong to Alice. Bob also performs a query for his own keys as well (so that his other devices can see what he sent).

Bob then encrypts the message using AES256-CFB-HMAC-SHA256 with a random key, encrypts the random key with each one of Alice's and Bob's public keys using RSA-OAEP, and forwards the message to Banana Inc to deliver. When queried, the keyserver not only knows what the query is for but also who is making the query.

The Federal Bulls**t Investigators (FBSI) are investigating Bob for Crimes against Humanity, namely playing Cards Against Humanity in the middle of Moffit Library while CS161 students were trying to study for their final. In their investigation, they want to wiretap Bob's communications.

(a) (8 points) The FBSI wants Banana Inc to provide the FBSI with a copy of all future communications sent by Bob by only modifying Banana Inc's server. Banana Inc complains that they can't, because to do so would require modifying Bob's bMessage client, and Bob does not believe in applying software updates. Who is correct? **Mark ONE of the following** and **BRIEFLY explain** (it should fit in a tweet) your answer.

○ Banana Inc ● FBSI

Tweet Length Explanation:

Solution: You modify the keyserver so that when Bob looks up a key, it also returns a key for the FBSI.

(b) (6 points) The FBSI has the ability to ask Banana Inc for a copy of all of Bob's ENCRYPTED bMessages, but without the key they can't decrypt them. The FBSI, tired of Banana's intransigence, instead decides to get access to Bob's key another way, by hacking. (They are the Law, so it's "legal").

Banana recently released a web-browser based client which allows Bob to access his bMessages from any web browser, bMessage-Web, a feature Bob decides he likes to use. When it is first run, the bMessage web-client's JavaScript creates a random 2048b RSA key and a password (which is *different* from the password used to log onto the bMessage web site) from Bob. It then creates a key using PBKDF2-SHA256, using this key with AES256-CFB-HMAC-SHA256 to encrypt the private RSA key for storing on Banana Inc's web server. This random key becomes another key for Bob.

Now on future logins, the JavaScript gets the encrypted RSA private key from the server (where it is stored in a SQL database), decrypts it using Bob's password, and then uses it to decrypt Bob's bMessages. And Bob is, well, a typical user and his passwords in general have 50b of entropy or less.

Assume there exists an XSS vulnerability. Provide a tweet-length strategy where the FBSI can use a XSS vulnerability in bMessagesWeb to probably access bob's messages.

Solution: You inject JavaScript onto the page to send Bob's private key to the FBSI.

(c) (6 points) Assume there exists an SQLI vulnerability. Provide a tweet-length strategy where the FBSI can use a SQLI vulnerability in bMessagesWeb to probably access Bob's messages.

Solution: You get the encrypted blob and do a brute force attack on the password

(d) (6 points) Assume that the FBSI has obtained Bob's website password but not his bMessage password. Provide a tweet-length strategy where the FBSI can use Bob's website password (NOT his bMessage password) to get his messages.

Solution: You get the encrypted blob and do a brute force attack on the password

(e) (4 points) Banana Inc is sick of the FBSI hacking, so they add a network-based Intrusion Prevention System that is capable of monitoring all requests to the bMessage web server (since the web server tells the NIDS about the keys for all TLS connections) and blocking all requests that don't satisfy the criteria. Which of the previous FBSI attacks could this host-based system potentially stop? Select all which apply:

- ● The XSS attack
- ● The SQLI attack
- ○ The FBSI has Bob's web password
- ○ None of the above

Problem 5 ***A Tour of Tor*** **(24 points)**

As a reminder, when connecting to a normal website through Tor, your computer first queries the Tor "consensus" to get a list of all Tor nodes, and using this information it connects to the first Tor node and, from their, creates a circuit through the Tor network, eventually ending at an exit node.

(a) (4 points) Consider the scenario where you are in a censored country and the censor choses not to block Tor, the censor is the adversary, and no Tor relays exist within this country. How many Tor relays must your traffic pass through, including the exit node, to prevent the censor from blocking your traffic.

- ● One
- ○ Two
- ○ Three
- ○ Four
- ○ Tor doesn't stop this adversary

(b) (4 points) Consider the scenario where you are the only user of Tor on a network that keeps detailed logs of all IPs contacted. You use Tor to email a threat. The network operator is made aware of this threat and that it was sent through Tor and probably originated on the operator's network. How many Tor relays must your traffic pass through, including the exit node, to guarentee the network operator can't identify you you as the one who sent the threat?

- ○ One
- ○ Two
- ○ Three
- ○ Four
- ● Tor doesn't stop this adversary

(c) (4 points) Consider the scenario where there is a single hostile Tor node but you don't know that node's identitity, and that node can be an exit node. You want to keep confidential from this node what HTTP sites you are visiting through Tor. How many Tor relays must your traffic pass through, including the exit node, to guarentee this adversary can't know what sites you visit?

- ○ One
- ● Two
- ○ Three
- ○ Four
- ○ Tor doesn't stop this adversary

(d) (4 points) Consider the scenario where there are mulitple independent hostile Tor nodes but you don't know that nodes identitities, and these nodes can be a exit nodes. You want to keep confidential from all these nodes what HTTP sites you are visiting through Tor. How many Tor relays must your traffic pass through, including the exit node, to guarentee that every independent hostile node can't know what sites you visit?

- ○ One
- ○ Four
- ● Two
- ○ Tor doesn't stop this adversary
- ○ Three

(e) (4 points) Consider the scenario where there are mulitple colluding hostile Tor nodes but you don't know those nodes identitities, and these nodes can be a exit nodes. You want to keep confidential from all these nodes what HTTP sites you are visiting through Tor. How many Tor relays must your traffic pass through, including the exit node, to guarentee that the colluding system of hostile nodes can't know what sites you visit?

- ○ One
- ○ Four
- ○ Two
- ● Tor doesn't stop this adversary
- ○ Three

(f) (4 points) Consider the scenario where there is a single hostile Tor node but you don't know that node's identitity, and that node can be an exit node. You want to have data integrity for the HTTP sites you are visiting through Tor. How many Tor relays must your traffic pass through, including the exit node, to guarentee this adversary can't manipulate the data you receive from the sites you visit?

- ○ One
- ○ Four
- ○ Two
- ● Tor doesn't stop this adversary
- ○ Three

Problem 6 ***Distributed Web Engineering*** **(24 points)**

In the old days, web sites used to run on just a single computer. But now, a modern web "site" can be spread over multiple computers and even multiple domains. So a user's login "cookie" needs to work transparently. All the servers which make up the site share a common 256b secret S and a common secret key K_s. The login server has a public key K_l.

Consider some of the following schemes. In these, the adversary has multiple accounts to experiment with and their goal is to be able to create a fake cookie for a targeted account they don't control.

(a) (8 points) In scheme 1, when you login, your browser presents your password P to the login server. The login server sets your *login* cookie as $Name||expires$ and sets an additional *auth* cookie as $Sign(K_l, login)$, using 512b RSA.

Can this prevent an attacker from making fake cookies?

Mark ONE of the following and **BRIEFLY explain** (tweet-length) your answer.

○ Yes ● No

Tweet Length Explanation:

Solution: No: The reason is the key length is just simply too small

(b) (8 points) In scheme 2, when you login, your browser presents your password P to the login server. The login server sets your *login* cookie as $Name||expires$ and then has JavaScript in the browser create the *auth* cookie as $HMAC(K_s, login)$.

Can this prevent an attacker from making fake cookies?

Mark ONE of the following and **BRIEFLY explain** (tweet-length) your answer.

○ Yes ● No

Tweet Length Explanation:

Solution: The attacker can get S from the JavaScript!

(c) (8 points) In scheme 3, when you login, your browser presents your password P to the login server. The login server sets your *login* cookie as $Name||expires$ and then sets the *auth* cookie as $SHA256(S||login)$

Can this prevent an attacker from making fake cookies?

Mark ONE of the following and **BRIEFLY explain** (tweet length) your answer.

● Yes ○ No

Tweet Length Explanation:

Solution: This works. Without knowing S, can't create a valid hash.

Problem 7 ***DNSSEC*** **(18 points)**

The UC Berkeley administrators recently announced a change in policy. When you create a new `berkeley.edu` subdomain, DNSSEC is now disabled by default, even thought `berkeley.edu` does support DNSSEC and uses NSEC for proving domains don't exist.

The CS161 staff just created a subdomain `insecurity.berkeley.edu` and populated domains within it.

(a) (4 points) If a recursive resolver with an empty cache which does NOT support DNSSEC wants to look up `tragic.insecurity.berkeley.edu`, which of the following queries will it make (select all that apply).

- ● `tragic.insecurity.berkeley.edu` from a root
- ● `tragic.insecurity.berkeley.edu` from a nameserver for `.edu`
- ● `tragic.insecurity.berkeley.edu` from a nameserver for `berkeley.edu`
- ● `tragic.insecurity.berkeley.edu` from a nameserver for `insecurity.berkeley.edu`
- ○ DNSKEY for `.` from a root
- ○ DS for `.` from a root
- ○ DNSKEY for `edu` from a root
- ○ DS for `edu` from a root
- ○ DNSKEY for `edu` from a nameserver for `.edu`
- ○ DS for `edu` from a nameserver for `.edu`
- ○ DNSKEY for `berkeley.edu` from a nameserver for `.edu`
- ○ DS for `berkeley.eduedu` from a nameserver for `.edu`
- ○ DNSKEY for `berkeley.edu` from a nameserver for `berkeley.edu`
- ○ DS for `berkeley.eduedu` from a nameserver for `berkeley.edu`
- ○ DNSKEY for `insecurity.berkeley.edu` from a nameserver for `berkeley.edu`
- ○ DS for `insecurity.berkeley.edu` from a nameserver for `berkeley.edu`

(b) (4 points) If a recursive resolver with an empty cache which does support DNSSEC wants to look up `tragic.insecurity.berkeley.edu`, which of the following queries will it make (select all that apply).

- [x] `tragic.insecurity.berkeley.edu` from a root
- [x] `tragic.insecurity.berkeley.edu` from a nameserver for `.edu`
- [x] `tragic.insecurity.berkeley.edu` from a nameserver for `berkeley.edu`
- [x] `tragic.insecurity.berkeley.edu` from a nameserver for `insecurity.berkeley.edu.edu`
- [x] DNSKEY for `.` from a root
- [ ] DS for `.` from a root
- [ ] DNSKEY for `edu` from a root
- [x] DS for `edu` from a root
- [x] DNSKEY for `edu` from a nameserver for `.edu`
- [ ] DS for `edu` from a nameserver for `.edu`
- [ ] DNSKEY for `berkeley.edu` from a nameserver for `.edu`
- [x] DS for `berkeley.edu` from a nameserver for `.edu`
- [x] DNSKEY for `berkeley.edu` from a nameserver for `berkeley.edu`
- [ ] DS for `berkeley.edu` from a nameserver for `berkeley.edu`
- [ ] DNSKEY for `insecurity.berkeley.edu` from a nameserver for `berkeley.edu`
- [x] DS for `insecurity.berkeley.eduedu` from a nameserver for `berkeley.edu`

(c) (4 points) If a validating recursive resolver looks up `horrible.insecurity.berkeley.edu` it gets an NXDOMAIN (this name doesn't exist) response. What crypotgraphic assertions can the resolver make about this NXDOMAIN error.

- [x] `insecurity.berkeley.edu` does not support DNSSEC
- [x] Nobody has tampered with the repies from the `berkeley.edu` nameserver
- [ ] `horrible.insecurity.berkeley.edu` does not exist
- [ ] Nobody has tampered with the repies from the `insecurity.berkeley.edu` nameserver

(d) (6 points) If all Berkeley services are mandated to use only end-to-end encrypted protocols, does the removal of DNSSEC have a practical impact on security? **Mark ONE of the following** and **BRIEFLY explain** ($\leq$ 2 sentences) your answer.

○ Yes ○ No

Tweet Length Explanation:

Solution: No significant effect because an end-to-end protocol doesn't acutally depends on correct naming.

Problem 8 ***Securing the Vault*** **(40 points)**

BearBank stores all of its sensitive company information on a set of "air-gapped" machines (i.e., no Internet connection). These machines are locked inside of a large vault with one door that contains a badge scanner (S). Employees at BearBank carry authentication badges that support all of the cryptographic primitives discussed in class.

After taking CS 161, Frodo is hired by BearBank to design a secure protocol for checking whether an employee is authorized to enter the vault. In particular, BearBank would like Frodo's system to protect against "skimming" attacks. In a skimming attack, an attacker (Mallory) knows all of the details about Frodo's authentication protocol, except secret key values; Mallory will interact with a victim on one day and then conduct an attack **the next day**. Specifically, on day #D, Mallory tricks an authorized user (Bob) into scanning his badge (P) on her malicious scanning device. This malicious device and P engage in Frodo's authentication protocol multiple times, and all of P's responses are recorded. The malicious device can spoof messages that look like what the real scanner (S) would send as long as the messages don't require a secret key to generate. Finally, on **the next day** (Day $\#D+1$), Mallory tries to gain access to the vault after analyzing the responses recorded from P.

Unfortunately, Frodo skipped CS 161 discussion sections, so he's not sure whether any of the following three designs are truly secure. For the three subparts below, select whether the protocol is secure or insecure. If the protocol prevents skimming attacks while still allowing authorized users to access the vault, then the protocol is secure. A protocol is insecure if either authorized users cannot access the vault or if Mallory can gain access to the vault. Justify your answer in 1-2 sentences: If the protocol is secure, explain what specific cryptographic primitive(s) guarantee its security and why. If the protocol is insecure, briefly describe an attack, or why legitimate users wouldn't be able to access the vault.

(a) (8 points) Authentication Protocol:

1. For each authorized user, P contains an RSA key pair pk, sk, where pk is the public key and sk is the private key. The vault scanner S stores a copy of each authorized user's pk and has access to an accurate clock.
2. When P is scanned by S, it sends U (the user's name) to S.
3. S then generates and stores $N =$ a new, random 128-bit string for the user and sends N to P. Since a legitimate badge will complete this entire protocol in under 1 minute, if the user doesn't attempt to authenticate within 2 minutes, S deletes N for that user and will randomly generate a new N during the user's next access attempt.
4. P signs N using its private key: $X = N_{sk}$, and sends (X, U) to S.
5. S checks that X is a legitimate signature for U on the N that it randomly generated for U. If the signature is valid, S allows the user to enter the vault and deletes N for the user. Otherwise, access is denied.

● Secure ○ Insecure

Solution: Secure. Standard challenge response: N is randomly generated and deleted after a purchase, so replay attacks are not possible. An attacker can only generate valid X's in the future if they can break RSA signatures. Might need to clarify the wording to prevent lame attacks where the skimmer just issues a fraudulent charge during the initial interaction; i.e., the model I want is that the protocol should prevent the skimmer from gaining information that allows them to make up new bills after the interaction w/ user.

(b) (8 points) Authentication Protocol:

1. For each authorized user, P contains a unique 128-bit symmetric key k, and S stores a copy of k for each user and has access to an accurate clock.
2. When P is scanned by S, it sends U (the user's name) to S.
3. S gets the current time from its clock: T, rounded to the neares 30 seconds, and sends T to P.
4. P computes an HMAC of T with k: $X = HMAC(T, k)$, and sends (X, T, U) to S.
5. S checks that X is a valid HMAC on T for U. Additionally, Since a legitimate badge will complete this entire protocol in under 1 minute, S also checks that T is within the past two minutes of the current time on its clock. If both these checks pass, then BearBank allows the user to access to vault. Otherwise, access is denied.

○ Secure ● Insecure

Solution: Insecure. The malicious machine can just ask P to compute HMAC's on a bunch of *future* timestamps, and then time things correctly to make fraudulent charges.

(c) (8 points) Authentication Protocol:

1. For each authorized user, P contains a unique random 128b secret key K_u and a highly accurate clock. S stores a copy of each user's k
2. When P is scanned by S, it sends U (the user's name) and $HMAC(S, time)$, with the current time rounded to the nearest 30 seconds to S.
3. S then checks if $HMAC(K_u, time)$ for both the current time (rounded to 30 seconds), and the rounded time +/- 30 seconds.

● Secure ○ Insecure

Solution: Secure. This is basically an RSA security token type setup, attacker can't get K_u

Just in case an attacker ever manages to get into the vault and install malware onto some of the machines, BearBank has tasked Frodo with installing a secure detection system on all of the machines.

(d) (4 points) For one approach, Frodo is thinking of configuring the machines to only allow read-operations. If any program tries to modify or delete data on a machine, it will explode and trigger an alarm in the vault. Which of the following detection approaches does this best represent? You do not need to explain your answer for this part.

- ○ Vulnerability based.
- ○ Anomaly based.
- ○ Signature based.
- ○ Behavioral based.
- ● Specification based.
- ○ Honeypot based.

(e) (4 points) Another approach Frodo has come up with is to install a bunch of dummy machines inside of the vault. If a new program is ever installed on any of these dummy machines, it will explode and trigger and alarm in the vault. Which of the following detection approaches does this best represent? You do not need to explain your answer for this part.

- ○ Vulnerability based.
- ○ Anomaly based.
- ○ Signature based.
- ○ Behavioral based.
- ○ Specification based.
- ● Honeypot based.

(f) (8 points) Frodo has settled on a signature based HIDS that analyzes a program and achieves an 87% true positive rate and a 2% false positive rate for detecting whether a program is malware. However, Aragon, one of his co-workers has developed an anomaly based HIDS that analyzes a program and achieves a 95% true positive rate and a 5% false positive rate. Assume that one out of every ten-thousand programs that run on a machine in BearBank's vault is malware. The total cost of failing to detect a malware sample is $100,000 dollars, the total cost of a false positive is $1,000. Assuming these values hold over the long run, which detector should BearBank deploy? Select your answer from the multiple choice below and **explain your answer** in 1-2 sentences.

- ○ We cannot make this assessment since we do not know the base rate
- ○ Frodo's detector is better.
- ○ Aragon's detector is better.
- ○ They are equally good.
- ● Neither, they both cost too much.

Solution: In order to compare which detector is better, we need to know their FP and FN rates, as well as the relative cost of a FP and FN, AND the base rate of attacks to normal events. With this information, we can assess the cost of each detector in the long run:

But lets just do a lazy ballpark and ONLY look at the cost of no detector: No detector its $100,000 for 10k runs

For 10k runs, Frodo's false positive rate of 2% means 200 false positives, at $1000 each, or $200k for the same # of runs in false positives alone!

Problem 9 ***Name That Pwn*** **(15 points)**

For the following questions, some classic attacks are described. Identify which attack(s) they represent. Fill out all matching solutions.

(a) (3 points) My name is `Robert; drop table students;--`

- ○ Stored XSS
- ○ Reflected XSS
- ● SQLI
- ○ Clickjacking
- ○ Buffer Overflow
- ○ None of the Above

(b) (3 points) My name is `Robert <script src="https://www.evil.com/pwnme.js">`. Visit my Facebook page, OK?

- ● Stored XSS
- ○ Reflected XSS
- ○ SQLI
- ○ Clickjacking
- ○ Buffer Overflow
- ○ None of the Above

(c) (3 points) My name is `Robert ... gah thats long ... shellcode goes here....`

- ○ Stored XSS
- ○ Reflected XSS
- ○ SQLI
- ○ Clickjacking
- ● Buffer Overflow
- ○ None of the Above

(d) (3 points) Oh, Interesting web request... Lets inject a packet...

- ● NSA QUANTUM
- ○ PuppyKitty
- ● The Great Firewall
- ● Airpwn
- ○ Firesheep
- ○ None of the Above

(e) (3 points) Hey, Ken, lets modify the compiler so that when it compiles login it inserts a backdoor...

- ○ Stored XSS
- ○ Reflected XSS
- ○ SQLI
- ○ Clickjacking
- ○ Buffer Overflow
- ● None of the Above

There is no need to penetrate a network when you can breach the people who run it. Networks are hard. People are soft. -Taylor Swift