Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.893 Fall 2009

Quiz II

All problems are open-ended questions. In order to receive credit you must answer the question as precisely as possible. You have 80 minutes to finish this quiz.

Write your name on this cover sheet.

Some questions may be harder than others. Read them all through first and attack them in the order that allows you to make the most progress. If you find a question ambiguous, be sure to write down any assumptions you make. Be neat. If we can't understand your answer, we can't give you credit!

THIS IS AN OPEN BOOK, OPEN NOTES EXAM.

Please do not write in the boxes below.

I (xx/10)	II (xx/30)	III (xx/10)	IV (xx/10)
V (xx/10)	**VI (xx/20)**	**VII (xx/10)**	**Total (xx/100)**

Name:

I KeyKOS

1. [10 points]: Bob is running the privilege-separated Zoobar web site on a KeyNIX system, using code from lab 3. Suggest a way in which Bob can modify the Zoobar server-side code to take advantage of KeyKOS capabilities to improve the security of his site, in a way that he wouldn't be able to do on Linux.

II Network protocols

Bob logs into an Athena workstation, which uses Kerberos to obtain a ticket for bob@ATHENA.MIT.EDU, and then runs Bob's mail client, which contacts Bob's post office server to fetch new messages.

2. [10 points]: Alice doesn't want Bob to know about an upcoming event, which was announced to Bob via email. To this end, Alice plans to intercept Bob's communication with his post office server, and to pretend that Bob has no new mail. Alice can observe and modify all network packets sent by Bob. How does Kerberos prevent Alice from impersonating Bob's mail server? Be as specific as possible; explain how Bob can tell between Alice and the real mail server in terms of network packets.

Now, Alice wants to read Bob's email, and intercepts all network packets ever sent and received by Bob's workstation (which is the only computer that Bob uses). However, Alice does not know Bob's password to access Bob's post office server, and Bob's packets to and from the post office server are protected by Kerberos.

3. [10 points]: Suppose that after Bob reads and deletes all of his mail, Alice learns what Bob's password was. Describe how Alice can obtain Bob's past messages.

4. [10 points]: To prevent Alice from reading any more messages, Bob ensures that Alice cannot intercept any subsequent network traffic, and changes his Kerberos password. Could Alice still read Bob's mail after this? Explain why not or explain how.

III ForceHTTPS

5. [10 points]: Bob is developing a new web site, and wants to avoid the problems described in the ForceHTTPS paper. He uses HTTPS for all of his pages and marks all of his cookies "Secure". Assuming Bob made no mistakes, is there any reason for Bob's users to install the ForceHTTPS plugin and enable it for Bob's site? Explain why or why not.

IV BitLocker

6. [10 points]: Alice wants to make BitLocker run faster. She decides that computing a different IV_s for each sector (pg. 13 in the BitLocker paper) is needlessly expensive, and replaces it with the fixed value $E(K_{\text{AES}}, e(0))$ instead. Explain how an attacker may be able to leverage this change to obtain data from a stolen laptop that uses BitLocker in TPM-only mode.

V Tor

7. [10 points]: Bob is running a hidden service on top of Tor, and wants to know how frequently he should choose new introduction points. Bob cares about his identity not being exposed, and about the availability of his service. Help Bob make an informed choice by explaining the costs and benefits of rotating introduction points either more or less frequently.

VI BackTracker

8. [10 points]: Alice is using the VM-based BackTracker to analyze a compromised server, where an attacker obtained some user's password, logged in via SSH, exploited a buffer overflow in a setuid-root program to gain root access, and trojaned the login binary, all using high-control events that are still stored in the event logger. How can Alice figure out what *specific* vulnerability in the setuid-root program the attacker exploited? In what situations would this be possible or not possible?

9. [10 points]: The VM-based BackTracker system requires no modifications to the guest OS, but nonetheless makes assumptions about the guest OS. List assumptions that are critical to back-tracking attacks that used high-control events.

VII 6.893

We'd like to hear your opinions about 6.893, so please answer the following questions. (Any answer, except no answer, will receive full credit.)

10. [10 points]: If you could change one thing in 6.893, what would it be?

End of Quiz

6.858 Computer Systems Security

Fall 2014

Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.858 Fall 2010

Quiz II

All problems are open-ended questions. In order to receive credit you must answer the question as precisely as possible. You have 80 minutes to finish this quiz.

Write your name on this cover sheet.

Some questions may be harder than others. Read them all through first and attack them in the order that allows you to make the most progress. If you find a question ambiguous, be sure to write down any assumptions you make. Be neat. If we can't understand your answer, we can't give you credit!

THIS IS AN OPEN BOOK, OPEN NOTES EXAM.

Please do not write in the boxes below.

I (xx/32)	II (xx/5)	III (xx/9)	IV (xx/20)	V (xx/8)	VI (xx/6)	Total (xx/80)

Name:

I Backtracking Intrusions

Alice finds a suspicious file, `/tmp/mybot`, left behind by an attacker on her computer. Alice decides to use Backtracker to find the initial entry point of the attacker into her computer. In the following scenarios, would Alice be able to use Backtracker, as described in the paper, to find the entry point?

1. [2 points]: An attacker exploits a buffer overflow in a web server running as root, gets a root shell, and creates the `/tmp/mybot` file.

2. [2 points]: An attacker exploits a buffer overflow in a web server running as root, gets a root shell, and modifies the password file to create an account for himself. The attacker then logs in using the new account, and creates the `/tmp/mybot` file.

3. [2 points]: An attacker guesses root's password, logs in, and creates the `/tmp/mybot` file.

Ben Bitdiddle wants to use Backtracker on his web server running the Zoobar web application. Ben is worried about both SQL injection and cross-site scripting attacks, where an attacker might use the vulnerability to modify the profiles of other users.

4. [6 points]: Ben runs unmodified Backtracker on his server, and uses a known SQL injection vulnerability to test Backtracker, while other users are actively using the site. Ben finds that he cannot effectively track down the attacker's initial entry point, after he detects that one of the user's profiles has been defaced by the attack. Explain why Backtracker is not working well for Ben as-is.

5. [10 points]: Propose a modification of Backtracker that would allow Ben to find the attacker's initial entry point for SQL injection attacks. Be sure to explain how the log, EventLogger, and GraphGen would need to be modified for your design, if at all, and whether you need to add any additional logging components. It's fine if your design does not handle buffer overflow attacks, and only handles SQL injection.

6. [10 points]: Ben's modified Backtracker system still cannot catch the attacker's initial entry point for the Zoobar profile worm, which spreads through a cross-site scripting vulnerability. Propose a modified design for Backtracker that can track down the source of a XSS attack like the profile worm.

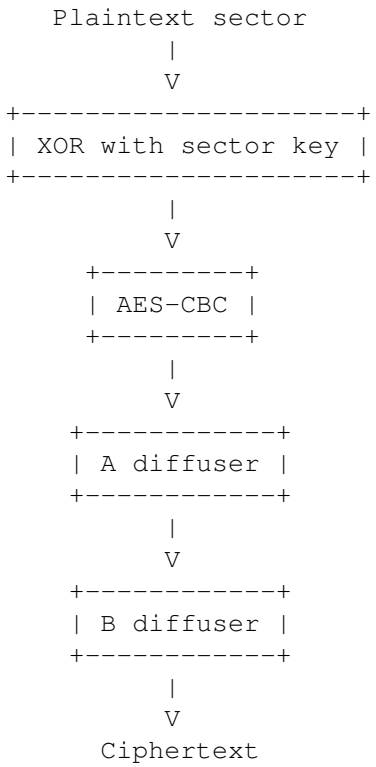
II TPMs

Suppose Ben implements Sailer's integrity measurement architecture on a Linux server, and a client uses the protocol in Figure 3 to verify the server's integrity (while sending these protocol messages over an SSL connection to Ben's server).

7. [5 points]: What, precisely, can a client safely assume about its SSL connection and about Ben's server, if it validates the response (steps 5a, 5b, and 5c)? Assume that no attackers have compromised any TPM chips or certificate authorities.

III Disk encryption

Ben Bitdiddle decides to optimize BitLocker's encryption mechanism by re-ordering some steps, so that the key-dependent sector key and AES-CBC steps can be combined. In particular, Ben's version of BitLocker looks like the follows (contrast with Figure 1 from the paper):



8. [9 points]: How can an adversary extract data from a stolen laptop running Ben's version of BitLocker in TPM-only mode, in a way that he or she could not for the original version of BitLocker? In other words, how is this scheme weaker?

IV Tor

Alice wants to improve the privacy of Tor, and changes the design slightly. In Alice's design, clients choose an exit node, and instead of building one circuit to the exit node, they build two circuits to the same exit node. Once the client builds both circuits, it sends the same randomly-chosen cookie to the exit node via each of the circuits, to tell the exit node that the two circuits belong to the same client. (After this point, the client and the exit node use the same stream IDs on both circuits interchangeably.) When a client wants to send a packet to the exit node, it sends the packet via one of the two circuits, chosen at random. Similarly, when the exit node wants to send data back to the client, it uses one of the two circuits at random.

9. [5 points]: What kinds of attacks against privacy does this scheme make more difficult?

10. [5 points]: What kinds of attacks against privacy does this scheme make easier?

11. [2 points]: What kinds of attacks against individual exit nodes does this scheme make easier?

12. [8 points]: Propose a modified design for Tor's hidden services that would allow a hidden service to require CAPTCHAs before spending resources on a client's request. Explain who generates the CAPTCHA in your design, who is responsible for checking the solution, and how the steps required to connect to a CAPTCHA-enabled hidden service change (along the lines of the list in Section 5.1 of the paper).

V IP Traceback

Ben Bitdiddle likes the IP Traceback scheme proposed by Stefan Savage, but doesn't like the fact that edge fragments are so small (consisting of just 8 bits of edge fragment data, as shown in Figure 9). Ben decides that he can get rid of the distance field, and expand the edge fragment field to 13 bits. To reconstruct the entire edge, Ben proposes to try all combinations of fragments (since he no longer knows what fragments came from the same distance away), at the cost of requiring more CPU time.

13. [8 points]: Describe a specific attack against Ben's scheme that violates the goal of IP Traceback (i.e., that the real path to the attacker is a suffix of the path returned by IP Traceback).

VI 6.858

We'd like to hear your opinions about 6.858, so please answer the following questions. (Any answer, except no answer, will receive full credit.)

14. [2 points]: What security topics did you want to learn more about, either in lectures or in labs?

15. [2 points]: What is your favorite paper from 6.858, which we should keep in future years?

16. [2 points]: What is your least favorite paper, which we should get rid of in the future?

End of Quiz

6.858 Computer Systems Security

Fall 2014

Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.858 Fall 2011

Quiz II

You have 80 minutes to answer the questions in this quiz. In order to receive credit you must answer the question as precisely as possible.

Some questions are harder than others, and some questions earn more points than others. You may want to skim them all through first, and attack them in the order that allows you to make the most progress.

If you find a question ambiguous, be sure to write down any assumptions you make. Be neat and legible. If we can't understand your answer, we can't give you credit!

Write your name on this cover sheet.

THIS IS AN OPEN BOOK, OPEN NOTES EXAM.

Please do not write in the boxes below.

I (xx/16)	II (xx/12)	III (xx/8)	IV (xx/8)	V (xx/8)
VI (xx/22)	**VII (xx/10)**	**VIII (xx/10)**	**IX (xx/6)**	**Total (xx/100)**

Name:

Username from handin site:

I Android

To help people keep track of todo items on their Android phone, Ben Bitdiddle writes *Ben's Todo Manager*. Ben wants to allow other applications to access todo items stored by his todo manager, so he implements a public *content provider* component that stores all of the todo items. To protect the todo items, Ben's application defines two new permissions, `com.bitdiddle.todo.read` and `com.bitdiddle.todo.write`, which are meant to allow other applications to read and modify todo items, respectively. Ben also sets the read and write labels of his todo content provider to these two permissions, respectively.

1. [4 points]: What permission type ("normal", "dangerous", or "signature") should Ben choose for the two new permissions declared by his application? Explain why.

2. [4 points]: Ben decides to implement a notification feature for todo items: a day before a todo item is due, Ben's application sends a broadcast intent containing the todo item, notifying other applications so that they may tell Ben to start working on that todo item. Ben sends the broadcast intent by using the `sendBroadcast(intent)` function. Explain what security problem Ben may have created by doing so, and specifically how he should fix it.

3. [8 points]: Ben discovers that Android does not control which application declares which permission name. In particular, this means that another malicious application, installed before Ben's Todo Manager, could have already declared a permission by the name of `com.bitdiddle.todo.read`, and this will not prevent Ben's Todo Manager application from being installed. Explain the specific steps an adversary could take to attack Ben's application given this weakness.

II BitLocker

Recall that a trusted platform module (TPM) contains several platform configuration registers (PCRs). The `extend(n, v)` operation updates the value of PCR register n by concatenating the old value of that PCR register (call it x) with provided data, v, and hashing the result, i.e.,

$$x' = H(x \| v)$$

where H is SHA-1, $\|$ is the concatenation operator, and x' is the new value of PCR register n.

Suppose that H were instead an insecure hash function that admitted a *preimage attack*, that is, given some value a it is easy to find another value $b \neq a$ for which $H(a) = H(b)$, and with high probability, it's easy to find such a b that starts with a specific prefix.

4. [6 points]: Could an attacker who has stolen a computer defeat BitLocker protection on its hard drive, with high probability? Explain how, or argue why not.

5. [6 points]: Could an attacker who has stolen the hard drive, but not the computer, defeat BitLocker protection on that drive, with high probability? Explain how, or argue why not.

III Side channel attacks

Ben Bitdiddle wants to secure his SSL server against RSA timing attacks, but does not want to use RSA blinding because of its overhead. Instead, Ben considers the following two schemes. For each of the schemes, determine whether the scheme protects Ben's server against timing attacks, and explain your reasoning.

6. [4 points]: Ben proposes to batch multiple RSA decryptions, from different connections, and have his server respond only after all the decryptions are done.

7. [4 points]: Ben proposes to have the server thread sleep for a (bounded) random amount of time after a decryption, before sending the response. Other server threads could perform computation while this thread is asleep.

IV Tor and Privacy

8. [4 points]: An "Occupy Northbridge" protestor has set up a Twitter account to broadcast messages under an assumed name. In order to remain anonymous, he decides to use Tor to log into the account. He installs Tor on his computer (from a trusted source) and enables it, launches Firefox, types in `www.twitter.com` into his browser, and proceeds to log in.

What adversaries may be able to now compromise the protestor in some way as a result of him using Tor? Ignore security bugs in the Tor client itself.

9. [4 points]: The protestor now uses the same Firefox browser to connect to another web site that hosts a discussion forum, also via Tor (but only after building a fresh Tor circuit). His goal is to ensure that Twitter and the forum cannot collude to determine that the same person accessed Twitter and the forum. To avoid third-party tracking, he deletes all cookies, HTML5 client-side storage, history, etc. from his browser between visits to different sites. How could an adversary correlate his original visit to Twitter and his visit to the forum, assuming no software bugs, and a large volume of other traffic to both sites?

V Security economics

10. [8 points]: Which of the following are true?

A. True / False To understand how spammers charge customers' credit cards, the authors of the paper we read in lecture (Levchenko et al) had to collaborate with one of the credit card association networks (e.g., Visa and MasterCard).

B. True / False The authors of the paper (Levchenko et al) expect it would be costly for a spammer to switch acquiring banks (for credit card processing), if the spammer's current bank was convinced to stop doing business with the spammer.

C. True / False The authors of the paper (Levchenko et al) expect it would be costly for a spammer to switch registrars (for registering domains for click support), if the spammer's current registrar was convinced to stop doing business with the spammer.

D. True / False If mail servers required the sending machine to solve a CAPTCHA for each email message sent, spammers would find it prohibitively expensive to advertise their products via email.

VI Trusted hardware

11. [8 points]: Ben Bitdiddle decides to manufacture his own TrInc trinkets. Each one of Ben's trinkets is a small computer in itself, consisting of a processor, DRAM memory, a TPM chip, a hard drive, and a USB port for connecting to the user's machine.

To make his trinket tamper-proof, Ben relies on the TPM chip. Ben's trinket uses the TPM to seal (i.e., encrypt) the entire trinket state (shown in Figure 1 in the TrInc paper) under the PCR value corresponding to Ben's trinket software. The TPM will only unseal (i.e., decrypt) this state (including counter values and K_{priv}) if the processor was initially loaded with Ben's software. When the trinket is powered off, the sealed state is stored on the trinket's hard drive.

Ben's simplified trinket does not implement the symmetric key optimization from TrInc, and does not implement the crash-recovery FIFO Q.

Assume Ben's software perfectly implements the above design (i.e., no bugs such as memory errors), and that the TPM, processor, and DRAM memory are tamper-proof.

How can an adversary break Ben's trinket in a way that violates the security guarantees that a trinket is supposed to provide?

Alice works for a bank that wants to implement an electronic currency system. The goal of the electronic currency system is to allow users to exchange *coins*. There is exactly one type of coin, worth one unit of currency. Alice's bank maintains one server that initially hands out coins. The system should allow user A to give user B a coin even if the two users are disconnected from the rest of the world (i.e., cannot talk to the bank or to any previous holders of that coin). Furthermore, it should be possible for user B to now give a coin to user C without having to contact anyone else. It should be impossible for user A to "double-spend", that is, to give the same coin to two different users.

12. [14 points]: Design an electronic currency system assuming each user has a TrInc trinket. Assume each trinket's public key is signed by the bank, that everyone knows the bank's public key, and that all trinkets are tamper-proof and trustworthy.

Explain three aspects of your design:

- What is the representation of a coin that a user has to store?
  (It's OK if this representation is not constant size.)
- How does user A send a coin to user B?
- What should user B do to verify that it has received a legitimate coin?

VII Usability

13. [10 points]: Alice's bank gives up on the trinket idea as being too costly. Instead, Alice is now designing a banking application for Android. She is worried that users of her banking application may be tricked into entering their bank account information into another look-alike application, because there's no reliable way for a user to tell what application he or she may be interacting with.

For example, there's no way for a user to look at the screen and tell what application is currently running. Even if a user initially runs on a legitimate banking application, a malicious application can start an activity right after that, and display an identical screen to the user. Finally, applications can use full-screen mode to completely replace the entire Android UI.

Propose a design in which users can safely enter their credentials into a banking application. Your proposed design can involve changes to the Android system itself. Unmodified existing Android applications *must* continue to work in your new design (though if you change the UI as part of your design, it's OK if the applications look slightly different as a result). It's fine to require sensitive applications (e.g., Alice's new banking application) to do things differently in your design.

VIII Zoobar

14. [4 points]: After turning in lab 5, Ben Bitdiddle remarks that it is strange that the browser allowed him to call the lab e-mail script in an `<img>` tag, and suggests that browsers should protect against this attack by refusing to load images from URLs that contain query strings. If Ben's proposal is implemented, can an adversary still use `<img>` tags to email user cookies after exploiting a cross-site scripting bug?

15. [6 points]: Ben is working on a Javascript sandboxing system similar to FBJS and lab 6, and he is worried that bugs in the Javascript parsing library that is used to rewrite code (e.g., Slimit) can make his sandbox insecure. Suppose that Ben's Javascript parsing library has some bug in the code that *parses* Javascript code into an AST. Can an adversary exploit such a bug to subvert the sandbox? Give a sketch of how, or explain why not.

IX 6.858

We'd like to hear your opinions about 6.858 to help us improve the class for future years. Please answer the following questions. (Any answer, except no answer, will receive full credit.)

16. [2 points]: What other topics would you have wanted to learn about, either in lectures or in labs?

17. [2 points]: What is your favorite paper from 6.858, which we should keep in future years?

18. [2 points]: What is your least favorite paper, which we should get rid of in the future?

End of Quiz

6.858 Computer Systems Security

Fall 2014

Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.858 Fall 2012

Quiz II

You have 80 minutes to answer the questions in this quiz. In order to receive credit you must answer the question as precisely as possible.

Some questions are harder than others, and some questions earn more points than others. You may want to skim them all through first, and attack them in the order that allows you to make the most progress.

If you find a question ambiguous, be sure to write down any assumptions you make. Be neat and legible. If we can't understand your answer, we can't give you credit!

Write your name and submission website username (typically your Athena username) on this cover sheet.

This is an open book, open notes, open laptop exam.
NO INTERNET ACCESS OR OTHER COMMUNICATION.

Please do not write in the boxes below.

I (xx/17)	II (xx/16)	III (xx/10)	IV (xx/18)	V (xx/24)	VI (xx/12)	VII (xx/6)	Total (xx/103)

Name:

Submission website username:

I RSA timing attacks / Tor

1. [9 points]: Answer the following questions based on the paper "Remote Timing Attacks are Practical" by David Brumley and Dan Boneh.

(Circle True or False for each choice.)

A. True / False Removing access to a high-precision clock on the server running Apache would prevent the attack described in Brumley's paper from working.

B. True / False Avoiding the use of the Chinese Remainder Theorem and the associated optimizations (i.e., performing computations mod p and mod q, instead of mod n) in OpenSSL would prevent the attack described in Brumley's paper from working.

C. True / False David Brumley's attack, as described in the paper, would work almost as well if the neighborhood of n values was chosen as $g, g-1, g-2, \ldots, g-n$ (as opposed to $g, g+1, g+2, \ldots, g+n$ as in the paper).

2. [8 points]:

Alyssa wants to learn the identity of a hidden service running on Tor. She plans to set up a malicious Tor OR, set up a rendezvous point on that malicious Tor OR, and send this rendezvous point's address to the introduction point of the hidden service. Then, when the hidden service connects to the malicious rendezvous point, the malicious Tor OR will record where the connection is coming from.

Will Alyssa's plan work? Why or why not?

II OAuth

After reading the "Empirical Analysis of OAuth" paper, Ben Bitdiddle wants to revise the OAuth protocol to avoid some of the pitfalls of using OAuth in a real system. For each of the following proposed changes, indicate whether the change would make the protocol more secure, less secure, or the same in terms of security, by writing **MORE**, **LESS**, or **SAME**, respectively. If the change affects the security of the protocol (or makes some pitfalls more likely), give a specific attack that is possible with the change, or that is prevented by the change. The changes are not cumulative between questions.

3. [8 points]: Ben removes **r** from steps 2 and 3 of the server-flow protocol as shown in Figure 1 of the "Empirical Analysis" paper. Now, instead of matching the supplied **r** against a list of allowed URL patterns for **i**, the IdP server keeps track of the redirect URL to use for each RP (identified by parameter **i**).

4. [8 points]: Ben combines steps 1, 3, and 5 to improve performance: the user enters their credentials, and the RP's Javascript code sends the credentials along with the Authz request to the IdP server.

III BitLocker

5. [10 points]: Suppose that an adversary steals a laptop protected with BitLocker in TPM mode, and wants to gain access to the data on the BitLocker-encrypted partition. The adversary discovers a buffer overflow in the BIOS code that can be exploited to execute arbitrary code by a specially-crafted USB drive plugged in during boot. How can the adversary gain access to the encrypted data? Be as specific as possible: what operations should the adversary's injected code perform, both on the main CPU and on the TPM?

IV TrInc

Alyssa P. Hacker is building FiveSquare, a peer-to-peer application in which friends share their locations with each other (not to be confused with FourSquare, which uses a central server). Alyssa's FiveSquare application runs on mobile phones, and works by directly connecting to a friend's mobile phone over TCP.

6. [18 points]:

Alyssa is worried that users will modify the FiveSquare application to report different locations to different friends. Supposing every mobile phone had a TrInc trinket, how could Alyssa use the trinket to ensure FiveSquare users cannot pretend to be in two different locations at the same time? Assume that every FiveSquare user corresponds to a fixed $\langle$trinket-id, counter-id$\rangle$ tuple.

Specifically, what should the counter value represent? How should a user update their location, and in particular, what arguments should they pass to `Attest(counter-id, new-counter-value, message-hash)`? How should a user check a friend's location, and in particular, what arguments should the friend pass to `Attest(..)`?

V Android

You are implementing the Koi Phone Agent for Android, based on the papers from lecture. You implement interaction between the application and the Koi phone agent using intents. Refer to Figure 1 in the Koi paper in the following questions.

7. [8 points]: What Android permissions does the Koi phone agent have to request access to in its manifest? Where are these permissions defined (i.e., whether these permission strings are dangerous, etc)?

8. [8 points]: What permissions does the application have to request access to in its manifest? Where are these permissions defined (i.e., whether these permission strings are dangerous, etc)?

9. [8 points]: How should the Koi agent, together with the phone's user, ensure that the only applications that can access to the Koi agent are the ones that the user trusts?

VI Zoobar

Alyssa is reviewing Ben's lab 6 code for sandboxing Javascript, and observes that she can invoke arbitrary code using the following Javascript code (which goes through Ben's sandbox rewriter):

```
var code = 'alert(5)';  // or any other code that will escape the sandbox
var a = function() { return '__proto__'; };
var b = -5;
var c = { toString: function() {
            b++;
            if (b > 0) { return 'constructor'; } else { return 'eval'; }
          }
        };
var d = a[c];
var e = d(code);
e();
```

10. [4 points]: What did Ben miss in his sandbox implementation?

11. [8 points]: Propose a fix for Ben's problem; be as specific as possible (i.e., code is best).

VII 6.858

We'd like to hear your opinions about 6.858. Any answer, except no answer, will receive full credit.

12. [2 points]: What was your favorite paper, which we should definitely keep in future years?

13. [2 points]: What was your least favorite paper, which we should drop in future years?

14. [2 points]: What topic did 6.858 not cover, but you think would be a good fit in future years?

End of Quiz

6.858 Computer Systems Security

Fall 2014

Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.858 Fall 2013

Quiz II

You have 80 minutes to answer the questions in this quiz. In order to receive credit you must answer the question as precisely as possible.

Some questions are harder than others, and some questions earn more points than others. You may want to skim them all through first, and attack them in the order that allows you to make the most progress.

If you find a question ambiguous, be sure to write down any assumptions you make. Be neat and legible. If we can't understand your answer, we can't give you credit!

Write your name and submission website email address on this cover sheet.

This is an open book, open notes, open laptop exam.
NO INTERNET ACCESS OR OTHER COMMUNICATION.

This quiz is printed double-sided.

Please do not write in the boxes below.

I (xx/14)	II (xx/8)	III (xx/9)	IV (xx/14)	V (xx/6)	VI (xx/18)	VII (xx/18)	VIII (xx/9)	IX (xx/4)	Total (xx/100)

Name:

Submission website email address:

This page intentionally left blank.

I Web security

1. [8 points]:

Recall that Zoobar has a cross-site scripting vulnerability that allows an adversary to construct a URL so that the server's response contains arbitrary Javascript code (from the URL itself). In lab 5, you used this vulnerability to steal a victim's cookie, by loading the `sendmail.php` script from our web server (`css.csail.mit.edu`).

Ben Bitdiddle is building a web browser, and he comes up with a plan to stop cross-site scripting attacks like the one above. His idea is to add an extra HTTP header, set by the web server, that prevents the browser from making *any* cross-origin requests from that page—including `IMG` tags, `SCRIPT` tags, forms, links, etc. If this header is set, Ben reasons that a cross-site scripting exploit like the one above will be unable to get to `sendmail.php`, and will not be able to steal the victim's cookie.

Explain how an adversary can steal a victim's cookie in Zoobar (say, running at `zoobar.com`), given a cross-site scripting vulnerability, without making cross-origin requests (i.e., bypassing Ben's plan), without network or DNS attacks, and without exploiting any other vulnerabilities.

This page intentionally left blank.

2. [6 points]:

Consider the following Javascript code that returns twice the value of its argument `x`:

```
function double(x) {
  var y = x;
  var cmd = "y=y+" + x;
  eval(cmd);
  return y;
}
```

Propose a design that extends lab 6 to safely support `eval`, such as in the above example, without allowing an adversary to execute arbitrary Javascript code in the web browser. Your design should support invoking `eval` on strings that are computed at runtime and cannot be determined statically, as in the above example. Your design should support the same subset of Javascript in `eval` as in lab 6 itself.

Don't worry about writing out exact Javascript code. We are looking for a precise yet succinct description of how such a design would work—a few sentences.

This page intentionally left blank.

II File system encryption

Ben Bitdiddle is designing an alternative to BitLocker that performs encryption at the file system level instead of disk sector level. Ben's encryption scheme works at the level of inodes (i.e., files or directories). Each inode is structured as follows:

```
struct inode {
  /* Encrypted parts of the inode */
  int inode_type;
  /* ... */
  uint64_t data_sectors[64];

  /* Non-encrypted parts of the inode */
  mac_tag inode_tag;
  mac_tag data_tag;
};
```

Each `struct inode` (except for the two tags at the end) is encrypted with AES-CBC using the master encryption key K and an IV of $E_K(inum)$, where *inum* is the inode number. The data sectors that comprise a file or directory are pointed to by the `data_sectors` array (which stores up to 64 sector numbers, in this simplified design). The contents of each data sector is encrypted with AES-CBC using the master encryption key K and an IV of $E_K(offset)$, where *offset* is the offset of that sector's data in the containing file or directory.

The inode's `inode_tag` is a MAC tag over the contents of the encrypted part of the inode, using the master authentication key A. The `data_tag` is a MAC over the contents of all data sectors of that inode, using the master authentication key A.

3. [8 points]:

Suppose an adversary obtains a laptop running Ben's encrypted file system, and cannot log in, but the laptop's software allows the adversary to create temporary files with arbitrary contents. How could the adversary gain access to Ben's data? Assume the adversary knows the inode number and sector numbers for any system file, such as `/etc/passwd`, since they are typically predictable. The adversary can power off the laptop at any time, take out the disk, and inspect/modify raw disk contents.

This page intentionally left blank.

III Network security

According to Mark Silis, if a firewall was deployed at MIT, would it make the following system less vulnerable to attacks, more vulnerable to attacks, or the same? Circle the best answer for each question.

4. [3 points]: MIT's building control systems, in the context of remote buffer overflow attacks.

A. More vulnerable.

B. Less vulnerable.

C. Just as vulnerable.

5. [3 points]: MIT's web servers, in the context of DDoS attacks.

A. More vulnerable.

B. Less vulnerable.

C. Just as vulnerable.

6. [3 points]: MIT's domain name, in the context of attacks like the hijacking in January 2013.

A. More vulnerable.

B. Less vulnerable.

C. Just as vulnerable.

This page intentionally left blank.

IV User authentication

The Secure Remote Password (SRP) is a well-understood, public-domain client/server password scheme based on a challenge-response protocol that has the following nice properties:

- User and server authenticate each other without any other parties involved;
- Password is never sent over the wire;
- Password (or equivalent) is not stored at the server;
- Adversary who steals server data cannot masquerade as client; and
- Using the same password with two different servers results in different server-side data.

7. [4 points]:

Which of the security criteria from "The Quest to Replace Passwords" does SRP meet (meaning solid circle from the paper)? Answer "False" for open-circle (almost offers the benefit).

(Circle True or False for each choice.)

A. True / False S1 Resilient-to-Physical-Observation.

B. True / False S4 Resilient-to-Unthrottled-Guessing.

C. True / False S5 Resilient-to-Internal-Observation.

D. True / False S6 Resilient-to-Leaks-from-Other-Verifiers.

E. True / False S9 No-Trusted-Third-Party.

F. True / False S10 Requiring-Explicit-Consent.

G. True / False S11 Unlinkable.

Ben Bitdiddle is worried that PBKDF2 is not good enough for storing passwords in his web site, since it takes just 50 msec to compute the hash for a given password and salt combination. Instead, Ben comes up with the following scheme (in Python), which applies PBKDF2 in a chain to each letter in the password, using the previous letter's hash value as the salt (and using the initial salt for the first letter):

```
def store(pw, salt):
  result = [salt]
  for c in pw:
    h = pbkdf2.PBKDF2(c, result[-1]).hexread(32)
    result.append(h)
  return result

def check(pw, stored):
  if len(pw) != len(stored) - 1:
    return False
  for cpos, c in enumerate(pw):
    h = pbkdf2.PBKDF2(c, stored[cpos]).hexread(32)
    if stored[cpos+1] != h:
      return False
  return True
```

To store a password, Ben's web site stores the result of the `store()` function, and to check if a password supplied by some web browser is correct, Ben's web site passes it to the `check()` function, along with the previously stored hash of the user's password, and returns an error to the web browser if the password does not match (i.e., `check` returned `False`).

8. [10 points]: Explain how an adversary can break into any account on Ben's web site, which uses Ben's modified password scheme described on the previous page. Assume the adversary does **not** steal the stored passwords from the server.

This page intentionally left blank.

V Privacy

9. [6 points]:

Tor uses TLS (which provides confidentiality and integrity) between onion routers, and encrypts cells traversing these routers with AES (see Figure 1). In addition, Tor checks integrity at the edges of each stream. Why is it necessary to perform end-to-end stream integrity in addition to TLS between routers?

VI Android security

The manifest for the FriendTracker application is as follows:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="org.siislab.tutorial.friendtracker"
          android:versionCode="1" android:versionName="1.0.0">
    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".FriendTrackerControl" android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <provider android:authorities="friends" android:name="FriendProvider"
                  android:readPermission="org.siislab.tutorial.permission.READ_FRIENDS"
                  android:writePermission="org.siislab.tutorial.permission.WRITE_FRIENDS">
        </provider>

        <service android:name="FriendTracker" android:process=":remote"
                 android:permission="org.siislab.tutorial.permission.FRIEND_SERVICE">
        </service>

        <receiver android:name="BootReceiver">
            <intent-filter>
                <action android:name="android.intent.action.BOOT_COMPLETED"></action>
            </intent-filter>
        </receiver>
    </application>

    <permission android:name="org.siislab.tutorial.permission.READ_FRIENDS"></permission>
    <permission android:name="org.siislab.tutorial.permission.WRITE_FRIENDS"></permission>
    <permission android:name="org.siislab.tutorial.permission.FRIEND_SERVICE"></permission>
    <permission android:name="org.siislab.tutorial.permission.FRIEND_SERVICE_ADD"></permission>
    <permission android:name="org.siislab.tutorial.permission.FRIEND_NEAR"
                android:label="@string/permlab_friendNear"
                android:description="@string/permdesc_friendNear"
                android:protectionLevel="dangerous"></permission>
    <permission android:name="org.siislab.tutorial.permission.BROADCAST_FRIEND_NEAR"></permission>

    <uses-permission android:name="org.siislab.tutorial.permission.READ_FRIENDS"></uses-permission>
    <uses-permission android:name="org.siislab.tutorial.permission.WRITE_FRIENDS"></uses-permission>
    <uses-permission android:name="org.siislab.tutorial.permission.FRIEND_SERVICE"></uses-permission>
    <uses-permission android:name="org.siislab.tutorial.permission.FRIEND_NEAR"></uses-permission>
    <uses-permission android:name="org.siislab.tutorial.permission.BROADCAST_FRIEND_NEAR"></uses-permiss
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"></uses-permission>
    <uses-permission android:name="android.permission.READ_CONTACTS"></uses-permission>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"></uses-permission>
</manifest>
```

10. [6 points]: The manifest, shown on the previous page, states several permissions that FriendTracker needs (e.g., the `[...].ACCESS_FINE_LOCATION`). `[...].ACCESS_FINE_LOCATION` is a dangerous permission. What or who decides whether FriendTracker should be allowed to have this dangerous permission?

11. [6 points]: What permissions should the manifest of the FriendViewer application request to be able to write to the `friends` provider in the FriendTracker application?

12. [6 points]: Consider the setup in Figure 2 of the Android paper from lecture. Where else, in addition to the shown manifest, does the `BROADCAST_FRIEND_NEAR` permission show up? Be specific: which applications, where in those applications (code or manifest), and why does `BROADCAST_FRIEND_NEAR` show up there?

This page intentionally left blank.

VII CryptDB

The proxy rewrites SQL queries to perform them over encrypted columns stored on server. An encrypted column may be an onion with several layers of encryption. Assume that the server has a `employee` table with three columns: `name`, `SSN`, and `salary`, and that all three are encrypted using an onion.

13. [6 points]: To what query does the proxy rewrite the query `select SUM(salary) from employee where name="Alice"`? (If CryptDB removes any onion layers, please explain which onion layers are removed.)

14. [6 points]: Even with several onions, CryptDB cannot rewrite the query `select * from employee where salary + ssn > 50`. Explain why and propose an extension of CryptDB that would be able to execute it.

This page intentionally left blank.

15. [6 points]: Recall that CryptDB's design as described in the paper uses three different onions for each encrypted column (each with different layers). Consider an alternative design which uses a single onion with three layers: OPE (innermost), DET (middle), and HOM (outermost). For each of the three operations (+ or `SUM`; =; and >), explain whether they can be performed using this onion, and why. If some operation cannot be performed with this alternative design, but *can* be performed with CryptDB, give an example query.

VIII Code obfuscation

16. [9 points]:

For each of the following techniques used in the "Looking inside Dropbox" paper, would the technique be useful in de-obfuscating a web application written in Javascript, given the Javascript source code running in the user's web browser?

(Circle True or False for each choice.)

A. True / False Reverse-engineering opcode remapping would be useful.

B. True / False Monkey-patching would be useful.

C. True / False Intercepting network requests before they are encrypted with SSL would be useful.

IX 6.858

We'd like to hear your opinions about 6.858. Any answer, except no answer, will receive full credit.

17. [4 points]: We have noticed that lecture attendance is noticeably lower than quiz attendance. Why do you think this is, and what do you think might make lectures worth attending?

End of Quiz

6.858 Computer Systems Security

Fall 2014

Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.858 Fall 2014

Quiz II

You have 80 minutes to answer the questions in this quiz. In order to receive credit you must answer the question as precisely as possible.

Some questions are harder than others, and some questions earn more points than others. You may want to skim them all through first, and attack them in the order that allows you to make the most progress.

If you find a question ambiguous, be sure to write down any assumptions you make. Be neat and legible. If we can't understand your answer, we can't give you credit!

Write your name and submission website email address on this cover sheet.

This is an open book, open notes, open laptop exam.
NO INTERNET ACCESS OR OTHER COMMUNICATION.

This quiz is printed double-sided.

Please do not write in the boxes below.

I (xx/16)	II (xx/24)	III (xx/10)	IV (xx/16)	V (xx/8)	VI (xx/12)	VII (xx/8)	VIII (xx/6)	Total (xx/100)

Name:

Submission website email address:

I User authentication

1. [8 points]: A company named Vault wants to offer a secure, cloud-based backup system. When the user updates a local file, her Vault client opens a TCP connection to a Vault server, and uses the Diffie-Helman protocol to establish a secret symmetric key K with the server. Then, the client generates this string s:

$$s = \langle \text{ documentName, documentContent, userName, userPassword, randomNumber } \rangle$$

and sends the following message to the Vault server:

$$E_K(s, \text{HMAC}_K(s))$$

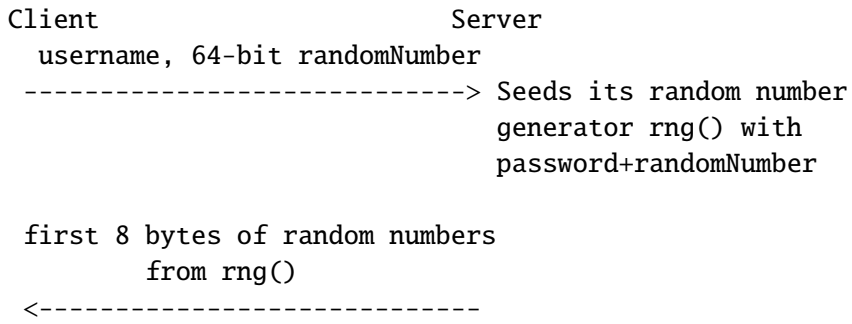
where $E_K(m)$ denotes encrypting message m using key K, and $\text{HMAC}_K(m)$ denotes computing an HMAC message authentication code of message m using key K.

The server decrypts the message, verifies the user's password, and verifies the integrity of the message using the HMAC. If all of the checks succeed, the server stores the document. If the server sees more than 10 messages with the wrong password, all future accesses to that account are blocked.

How can a network attacker reliably obtain the user's password?

2. [8 points]:

Suppose that a user wants to verify that a server knows the user's password. The user and the server engage in the following challenge/response:



Everyone knows which algorithm the server uses for `rng()`, so the client can validate that the server's response contains the expected bytes.

Suppose that many different servers use this protocol. Further suppose that the attacker has a list of popular usernames, and a separate list of popular passwords. Explain how an attacker can abuse the protocol (without otherwise compromising any servers) to build a rainbow table of passwords and easily determine the passwords of many users.

II Tor and Private browsing

Imagine that the website `https://foo.com` embeds a JavaScript file from `http://attacker.com`. Suppose that a user's browser allows an HTTPS page to run JavaScript code fetched from an HTTP URL; however, the code cannot read or write any persistent client-side state. For example, the JavaScript code cannot read or write cookies, nor can it read or write DOM storage. The browser also ensures that the `attacker.com` web server cannot set client-side cookies using the `Set-Cookie` header, or receive client-side cookies in the HTTP request for the JavaScript file.

Suppose that the user visits `https://foo.com` once in private browsing mode, closes the tab, and then visits the site again in regular browsing mode; in both cases, the user's web traffic goes through Tor. The `attacker.com` web server would like to determine with high likelihood that the same user has visited the site twice. However, the attacker does not control any Tor nodes.

3. [8 points]: Why is it unlikely that the `attacker.com` server can use TCP fingerprinting to identify the user? Recall that TCP fingerprinting involves looking at TCP connection parameters, such as the initial TCP window size, TCP options, TCP flags, etc.

4. [8 points]: Describe a way that the attacker can fingerprint the user with a much higher likelihood of success.

5. [8 points]: Browsing the web through Tor can be slow. This is because user traffic is forwarded between volunteer computers that are scattered across the world, overburdened with traffic, and potentially situated behind slow network connections.

Suppose that CloudCo, a large technology company with datacenters scattered across the world, offers some of its machines to the Tor community for use as entry and exit nodes. CloudCo machines have plentiful RAM and CPU; CloudCo machines also have low latency, high-bandwidth network connections to all major ISPs. By using CloudCo machines as Tor entry and exit nodes, users could ensure that Tor congestion would only exist in the middle of a circuit.

Assume that CloudCo genuinely wants to help Tor users, and that CloudCo configures its machines to faithfully execute the Tor protocol. Why is it still a bad idea for users to employ CloudCo machines as entry and exit nodes?

III TaintDroid

6. [10 points]: TaintDroid defines various sources of taint, and a unique taint flag for each of those sources (e.g., `IMEI`, `PHONE_NUM`, `CAMERA`, etc.).

For the application code below, list the set of taint flags that each variable will have *after* each line of code has executed (for example, `"IMEI, CAMERA"` or `"PHONE_NUM"`). If a variable will not have any taint flags, write an Ø.

```
int x = android.getIMEI();                  x:_____________
int y = android.getPhoneNum();              y:_____________
int z;
if(x > 5000){
    z = 0;                                  z:_____________
}else{
    z = 1;                                  z:_____________
}

int arr[] = new int[2];                     arr:___________
arr[0] = x;                                 arr:___________
arr[1] = y;                                 arr:___________

char buf[] = android.getCameraPicture();    buf:___________
int val = buf[x%2];                         val:___________

x = 42;                                     x:_____________
```

IV Timing side channels

Consider the timing attack on OpenSSL RSA keys, described in the paper by Brumley and Boneh.

7. [8 points]: Suppose that OpenSSL was modified to never use Karatsuba multiplication (and instead always use "normal" multiplication), but was still using Montgomery multiplication as described in the paper. Would you expect the attack to still work with a few million queries? Explain why or why not.

8. [8 points]: Suppose that OpenSSL was modified to never use Montgomery multiplication, but was still using both Karatsuba and normal multiplication as described in the paper. Would you expect the attack to still work with a few million queries? Explain why or why not.

V Embedded device security

9. [8 points]: Ben Bitdiddle has a smartphone with an always-on voice recognition system, which runs any commands that it hears.

Alyssa P. Hacker wants to trick Ben's phone into running a command, but Ben turns off all wireless radios on the phone as a precaution to prevent Alyssa from breaking in over the network, and also keeps his phone in a locked sound-proof room in hopes of foiling Alyssa. After hearing Kevin Fu's guest lecture, Alyssa figures out how she can get Ben's phone to run a command of her choice, without breaking into Ben's room. What is Alyssa's plan?

VI Android security

Ben Bitdiddle is still excited about his phone's voice recognition system, and decides to set up a system that allows third-party applications to handle user voice commands (e.g., his music player might want to support a command like "next song", and his Facebook application might want to support a command like "post my current location on Facebook").

Ben's design runs on Android, and involves a single voice recognizer application that runs with access to the microphone. This voice recognizer transcribes whatever sounds it hears into ASCII text messages, and hands these messages off to third-party applications.

10. [12 points]: Describe a design for allowing the voice recognizer to securely send the transcribed messages to third-party applications, and allowing the third-party applications to securely receive them. Be as specific as possible; do not worry about software bugs. To help you structure your answer, consider the following:

First, what new permission labels do you propose to create? For each new permission label needed in your design, specify (1) its name, (2) who creates the label, and (3) what the type of the permission label is.

Continued on the next page.

Second, how should the voice recognizer securely send the transcribed messages? Describe (1) what component(s) the recognizer should have, (2) what permission(s) the application should ask for, (3) what label(s) should protect the recognizer's component(s), and (4) what other security-relevant steps the recognizer's code has to take.

Third, how should the third-party applications securely receive the messages? Describe (1) what component(s) each application should have, (2) what permission(s) the application should ask for, (3) what label(s) should protect the application's component(s), and (4) what other security-relevant steps the application code has to take.

VII Labs

11. [8 points]: Ben Bitdiddle's lab 6 code rewrites the following profile code:

```
var x = y[z];
```

into:

```
var sandbox_x = sandbox_y[sandbox_z];
```

Write down an exact piece of profile code that an adversary could use to call `alert(123);`.

VIII 6.858

We'd like to hear your opinions about 6.858. Any answer, except no answer, will receive full credit.

12. [6 points]: Now that you are almost done with 6.858, how would you suggest we improve the class in future years?

End of Quiz

6.858 Computer Systems Security

Fall 2014

6.858 Quiz 2 Review

Android Security

Haogang Chen

Nov 24, 2014

Security layers

Layer	Role
Reference Monitor	**Mandatory Access Control (MAC) for RPC:** enforce access control policy for **shared resources**
Java VM	**Memory safety:** (neither required nor trusted)
Linux Kernel	**Isolation:** apps run with different UIDs. (principals are apps, as opposed to users)

Basic architecture

- Apps are composed of **components**
- 4 types of components
  - **Activity:** UI, only one active at a time
  - **Service:** background processing, RPC server
  - **Content provider:** provides read/write RPC
  - **Broadcast receiver:** listen for notifications

Intent: RPC primitive

- Has 4 fields
  - **Component:** target
  - **Action:** opcode
  - **Data:** arguments
  - **Category:** for filtering
- The **reference monitor** checks sender's permission labels upon message delivery.

Permission labels

- **Application** defines permissions as string labels
  - `<permission name="com.android.phone.DIALPERM"></…>`
- Application asks for permissions in its **manifest**
  - `<use-permission name="com.android.phone.DIALPERM"></…>`
- Application assigns a **type** for each permission

Permission types

Type	Reference Monitor's grant policy
Normal	Silent check, no user interaction required. (no security guarantee for any serious use…)
Dangerous	Ask the user upon app installation. (useful when you want to interact with others' apps)
Signature	Silently grant to apps signed by the same developer. (useful when you only talk to your own apps)

Implicit and broadcast intent

- Implicit intent
  - Omit the "*target*" field; let Android figure out the receiver
  - Receivers declare interested *actions* and *categories* using **intent filters**
- Broadcast intent
  - Problem: how to ensure only *someone* gets the broadcast?
  - Solution: protected broadcast (not MAC)
    - Request for a permission when broadcasting `sendBroadcast(intent, "perm.FRIEND_NEAR")`

Summary

- **Permissions:** "*Who are allowed talk to me?*"
- **Permission types:** "*How to grant permissions to an app?*"
- **Intent filters:** "*What (implicit intent) do I want to see?*"
- **Protected broadcast:** "*Who are allowed to see my (broadcast) intent?*"

6.858 Quiz 2 Review

TaintDroid

Haogang Chen

Nov 24, 2014

Motivation

- Limitation of the reference manager
  - "*What resource can I access?*"
    - No guarantee on ***how the data is being used***.
      - E.g., a photo editor can silently upload your photo stream to its server
- TaintDroid: track information flow for sensitive data

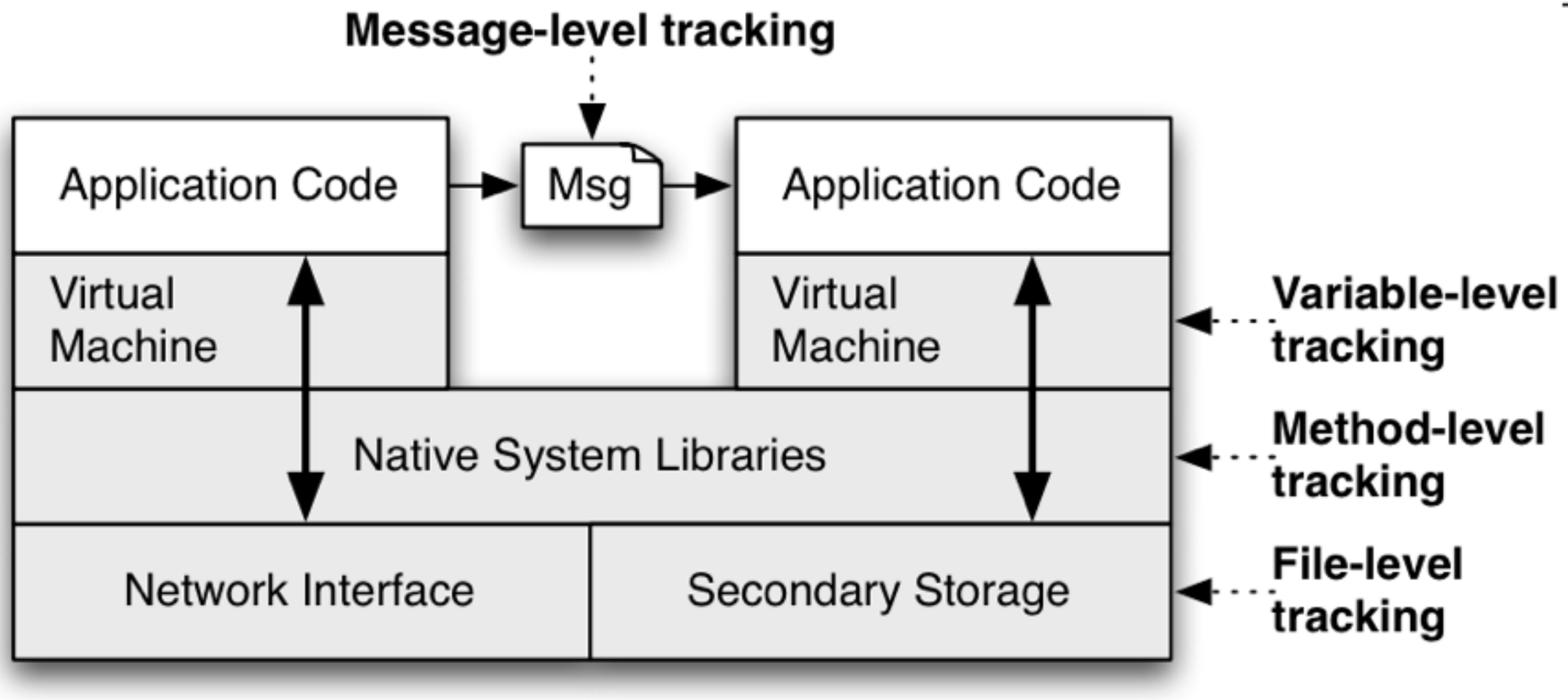
Taint tracking basics

- **Source:** origin of sensitive data
  - E.g., photos, contacts, GPS coordinates
- **Sink:** undesired destination
  - E.g., network interface, TCB boundary
- **Propagation**: how information flows from source to sink
  - E.g., variable copy, arithmetic operations, indexing, message passing, system calls, file read/write.

Approach

- Attach a “**tag**” for each piece of sensitive data
- Propagate the tag together with the data
- Challenges
  - Fine-grained tracking can be extremely slow
  - Coarse-grained tracking introduces false positives
  - Key contribution: trade-offs between performance and accuracy

TaintDroid: multi-level tracking



Component	Trusted?	Action
System app.	Y	**Taint source:** annotate data from sensitive content provider (e.g. camera app)
User app.	N	User apps runs inside Java VMs. They are untrusted and unmodified
Java VM	Y	**Variable-level tracking:** store and propagate taint tags in shadow memory for every variable
RPC library	Y	**Message-level tracking:** propagate taint tags when serializing/deserializing messages
System library	Y	**Method-level tracking:** annotate how taints propagate among arguments and return values
Storage library	Y	**File-level tracking:** attach and propagate taint tags in file's extended attribute.
Network library	Y	**Taint Sink:** annotate the interface, and report any tagged data that reaches the sink

Limitation of taint tracking

- Cannot capture control-flow dependencies

```
// “dirty” is tainted
int laundry(int dirty) {
    int clean;
    if (dirty == 0)
        clean = 0
    else if (dirty == 1)
        clean = 1
    else if (dirty == 2)
        clean = 2
    else …
    return clean;
}
```

6.858 Computer Systems Security

Fall 2014

User Authentication

- Know each factor that the paper used to judge authentication methods (i.e. Resilient-to-Physical-Observation)
- Be ready to explain why one authentication scheme would work better than another for some specified system

Private Browsing

- Two threat models
  - Attacker with your machine after you're done browsing
  - Attacker who has compromised a web server you've visited

Local Attack

- Attack surface includes:
  - Cookies, DOM storage
  - Browser cache
  - History of visited addresses
  - Configuration of browser
  - Downloaded files
  - New plugins/browser extensions
  - Persistent data in RAM

Remote Attack

- Attack surface includes:
  - IP address
  - Browser fingerprinting
  - Link Colors (attack has been depreciated)
  - Cookies

Browser State Bleeding

- Public state that bleeds into private state
  - Easier for remote attacker to link public and private sessions
- Private state that bleeds into public state
  - Makes job easier for both remote and local attacker
- Private state persisting in session
  - If private state does not persist, remote attacker can recognize private mode
- Private state persisting across session
  - Should not occur, but happens if there are state bleeds from private to public or public to private (certificates, downloads, etc)

6.858 Computer Systems Security

Fall 2014