CS 155: Spring 2014
June 9, 2014

CS 155 Final Exam

This exam is open book and open notes. You may use course notes and documents that you have stored on a laptop, but you may NOT use the network connection on your laptop in any way, especially not to search the web or communicate with a friend. **You have 2 hours.** Print your name legibly and sign and abide by the honor code written below. All of the intended answers may be written in the space provided. You may use the back of a page for scratch work. If you use the back side of a page to write part of your answer, be sure to mark your answer clearly.

The following is a statement of the Stanford University Honor Code:

A. *The Honor Code is an undertaking of the students, individually and collectively:*

   (1) *that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;*

   (2) *that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.*

B. *The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.*

C. *While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.*

I acknowledge and accept the Honor Code.

_____
(Signature)

_____ _____
(SUNet ID) *(Print your name,* legibly!*)*

☐ **GRADUATING?**

Prob	# 1	# 2	# 3	# 4	# 5	# 6	Total
Score							
Max	16	19	11	11	17	16	90

1. (*16 points*) ..... True or False

_____ (a) Stack canaries used to prevent stack smashing attacks can be applied to an executable without having to recompile the code.

_____ (b) DEP/NX defenses do not prevent return-oriented-programming (ROP) attacks.

_____ (c) System call interposition is a technique used to sandbox vulnerable applications.

_____ (d) Fuzzing will expose all vulnerabilities in an application.

_____ (e) A good technique for preventing CSRF attacks is session management using authenticators stored in cookies.

_____ (f) Both XSS and control hijacking attacks are caused by confusing the system into interpreting content as code.

_____ (g) SSL strip attacks can be defeated by using extended validation certificates.

_____ (h) A stateless packet-filter firewall can block incoming requests to an internal SMTP server, while allowing outgoing SMTP requests.

_____ (i) A stateful packet-filter firewall can block incoming requests to an internal SMTP server, while allowing outgoing SMTP requests.

_____ (j) Choosing random initial sequence numbers in the TCP handshake ensures that a network attacker cannot inject packets into the session.

_____ (k) HTTPS is designed to prevent a network attacker from replaying an old session.

_____ (l) Some DoS attacks would be possible even if every autonomous system on the Internet implemented ingress filtering.

_____ (m) Different iOS apps may be installed with different permissions, enforced by an application sandbox.

_____ (n) Different Android apps may be installed with different permissions, enforced by an application sandbox.

_____ (o) Different Windows Phone 8 apps may be installed with different permissions, enforced by an application sandbox.

_____ (p) Polymorphic viruses have a fixed header that makes them easily recognizable by virus checkers.

2. (*19 points*) ..... Short Answer

(a) (*4 points*) Explain how open DNS resolvers are used for DoS attacks. Why would an attacker use them?

(b) (*4 points*) Suppose the user has a browser tab open at `https://xyz.com/a.html` and the lock icon is properly shown in the address bar meaning that an HTTPS session was established without any errors. The user then opens a new tab pointing to `https://xyz.com/b.html`, but this time the browser receives an improper certificate for `xyz.com`. The browser warns the user, but the user ignores the warning and clicks through to get to the page. Clearly data on `b.html` is vulnerable to a network attacker mounting a man-in-the-middle attack. Can the network attacker steal data from the first tab pointing to `https://xyz.com/a.html`? If so, explain how. If not, explain why not. Please assume that the CA infrastructure is secure so that no rogue certificates for `xyz.com` have been issued.

(c) (*4 points*) List one advantage and one disadvantage of capabilities, compared to access control lists.

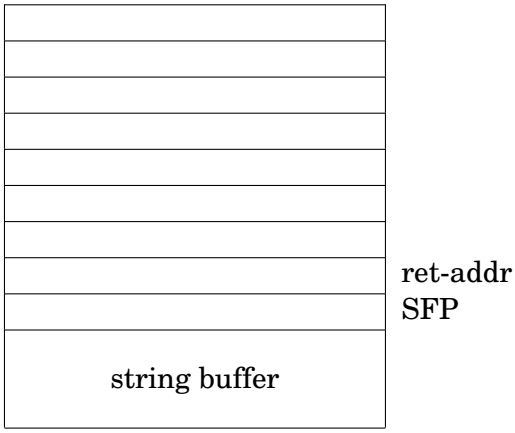
(d) (*4 points*) Return oriented programming. Suppose an attacker finds the following code fragments in the `libc` library once it is loaded in memory:

```
0x40002000:
  pop eax
  ret

0x40002700:
  pop ebx
  ret

0x40002948:
  move [ebx], eax     // write the value of eax to location ebx
  ret
```

Moreover, the attacker discovers a stack buffer overflow in some vulnerable application where a string buffer can overflow. Explain how in an environment where the stack is marked non-execute the attacker can nevertheless cause the vulnerable application to write the value `0x40000000` to memory address `0x41000000` and then jump to address `0x40044444`. Your answer need only show the contents of the stack in the table below right after the buffer overflow takes place.



(e) (*3 points*) Why does Android use both ProPolice and ASLR?

3. (*11 points*) ..... Upside-down stack

Under UNIX, as discussed in class, the run-time stack begins at a high memory address and grows "downward" in memory. In other words, when a function `f` is called, the address of the activation record for `f` is a lower address number than the activation record for the caller of `f`.

(a) (*4 points*) Explain why the order of activation records in memory, and the relative position of entries in an individual activation record, is relevant to buffer overflow attacks.

(b) (*7 points*) Are stack buffer overflow attacks still possible if the stack grows in the other direction? If so, give example vulnerable code and explain how it is vulnerable. If not, explain why not.

4. (*11 points*) ..... Browser malware

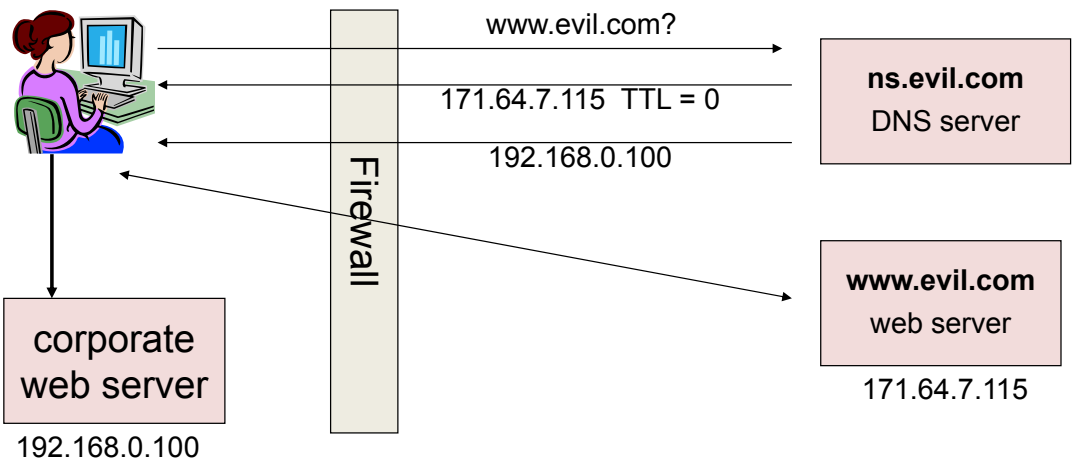
Browser extensions are Javascript apps that run in the browser whenever the browser is launched; these extensions typically interact with the user's browsing experience (e.g. spell-checking or dictionary lookup), but they can also make network connections, modify the DOM, etc. Users install extensions by downloading them from the browsers app store; the browser blocks an extension from running unless it was downloaded from the app store. Like Android apps, browser extensions need to request a set of permissions that restrict/allow what they can do; updates to browser extensions are allowed to request a new set of permissions. Some example permissions include redirecting user traffic, accessing a user's browsing history and bookmarks, and accessing/modifying the content (DOM) of web pages currently loaded in the browser. If an extension tries to use permissions it does not have, the browser blocks the extension from executing further.

Suppose the browsers security team searches for malicious extensions by running a test browser with a suspicious extension and visiting/interacting with popular websites. They use static and dynamic analysis techniques to determine if an extension is malicious:

(a) (*3 points*) Consider the following malicious extension: the entire extension code is encrypted except for a small decryption engine in the header that is available in the clear. Every time the extension runs, it connects to the author's server to get the decryption key and then decrypts and executes its code. In general, what does this obfuscation/encryption technique make less effective: static analysis or dynamic analysis? Justify your answer.

(b) (*4 points*) When testing an extension, the security team runs the extension on a test machine and monitors the extension's runtime behavior to see if it ever causes harm (e.g. sends out the user's cookies to `evil.com`). Suppose the extension's author knows the range of IP addresses assigned to the security team's test machines. Explain how the author can design a malicious extension to evade the security team's runtime monitoring, but steal a user's cookies when installed on a user's machine in the real world. Please assume the extension cannot read the IP address of the machine on which it is running.
Hint: Try extending the design from part (a).

(c) (*4 points*) To make scanning more efficient, whenever the author updates a benign extension, the security team checks if the code in any of the extension's methods/functions have changed. If none of the functions/methods were changed, the security team allows the extension to automatically update without user approval because it believes the update is benign (e.g. changing the extensions copyright text). Describe a security problem that this approach introduces.



5. (*17 points*) ..... DNS Rebinding Attacks

The figure above shows a browser behind a corporate firewall. In the first steps shown in the figure, the browser renders web content from `www.site.com` that contains an embedded frame `<iframe src="http://www.evil.com">`. Because the user has not requested a connection to `evil.com` before, the user's system asks the DNS server `ns.evil.com` for the IP address of `www.evil.com`. The DNS server returns a resource record (RR) giving the address as 171.64.7.115, with TTL = 0. The user's system uses the address 171.64.7.115 to load the frame.

A short time later, the same web content makes a second request to `www.evil.com` to load a second subframe of the same page. Because the TTL of the previous RR was 0, the user's system goes back to `ns.evil.com` and asks again for the IP address of `www.evil.com`. This time, the DNS server returns a resource record with the address 192.168.0.100, which happens to be the IP address of a corporate web server behind the firewall. This causes the user's browser to contact the corporate web server at 192.168.0.100.

(a) (*1 point*) For the same-origin policy governing DOM access in the browser, what are the three components of the origin?

(b) (*1 point*) What is the origin of the first frame loaded by the page?

(c) (*1 point*) What is the origin of the second frame loaded by the page?

(d) (*2 points*) Normally, content from `evil.com` cannot access DOM elements associated with content from the corporate web server. Explain how and why in the setup above content from `evil.com` can read content from the corporate web server.

(e) (*4 points*) One solution to the problem is called *DNS pinning*. In DNS pinning, the user's browser will not query DNS again. Instead, once a frame is loaded from `www.evil.com`, the same IP address will be used for all subsequent requests. Give one compelling disadvantage of DNS pinning?

(f) (*4 points*) Explain why DNSSEC does not prevent this attack.

(g) (*4 points*) Suppose you want to solve the problem at the firewall *without* blocking DNS traffic based on TTL. Give a firewall policy that will prevent this attack or explain why a firewall cannot prevent this attack.

6. (*16 points*) ..... Email Security

(a) (*4 points*) In the last lecture Markus Jakobsson discussed the DMARC email security mechanism. DMARC incorporates a mechanism called DKIM whereby every email sent by an organization like PayPal is digitally signed by PayPal. The recipient mail server retrieves PayPal's public key from PayPal's DNS record, checks the signature and rejects the email if the signature fails to verify. Explain why this system does not protect PayPal users from Phishing emails targeting PayPal.

(b) (*4 points*) When `alice@gmail.com` uses her Chrome browser to send an email to `bob@gmail.com` Alice's browser establishes an HTTPS connection to `gmail.com` and sends the email over this connection. The same happens when Bob reads the email in his Chrome browser. As discussed in class, Chrome *pins* the certificate for `gmail.com`. Can a network attacker read the email in the clear? Please explain. Recall that a network attacker only attacks the network links, but does not attack the end points (Alice, Bob, and Google). Moreover, the attacker is not part of Google nor does it collude with Google.

(c) (*8 points*) In the existing Gmail system the Gmail server sees all emails in the clear. Google recently announced a Chrome extension that provides end-to-end encryption for emails sent through Gmail. The system works as follows: (1) when first registering to use the system, Bob's Chrome extension generates a public/private key pair, (2) when Alice sends an email to Bob she first somehow obtains Bob's public key and then the Chrome extension encrypts the email body using Bob's public key before sending out the email, (3) when Bob retrieves the email the Chrome extension decrypts the encrypted contents using Bob's secret key.

Currently, Google's end-to-end encryption system cannot handle mailing lists. Your goal is to extend the Chrome extension so that Alice can send encrypted email to a mailing list. The system must satisfy the following properties: (1) only current members of the mailing list can read emails sent to the list, (2) the list server cannot see emails in the clear, (3) Alice does not know the individual members in the list; she only knows the list's email address and public-key, (4) the mailing list should provide a stable public key, that is a key that rarely changes.

In describing your system explain what the list manager does to enroll a new member, what the list manager does to remove a member from the list, and what the list server does when an incoming encrypted email is received. Note that the *list manager* is only responsible for managing group membership and does not see encrypted emails sent to the list. The *list server* is a separate entity that receives emails sent to the group and forwards them to group members.

Hint: You may assume that the public-key encryption system used has the property that any secret key SK can be randomly split into two keys SK_1 and SK_2. Decryption can proceed in two steps: first apply SK_1 to the ciphertext and then apply SK_2 to the result to obtain the cleartext content. Neither SK_1 nor SK_2 can be used to decrypt an incoming ciphertext on their own. Note that a single secret key SK can be repeatedly split into independent pairs $\left(\mathrm{SK}_1^{(1)}, \mathrm{SK}_2^{(1)}\right), \ldots, \left(\mathrm{SK}_1^{(n)}, \mathrm{SK}_2^{(n)}\right)$.