CS 155: Spring 2007
June 2007

CS 155 Final Exam

This exam is open books and open notes, but you may not use a laptop. You have 2 hours. Make sure you print your name legibly and sign the honor code below. All of the intended answers may be written well within the space provided. You may use the back of the preceding page for scratch work. If you want to 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)

☐ **SENIOR?**

_____

(Print your name, legibly!*)*

Prob	# 1	# 2	# 3	# 4	# 5	# 6	# 7	# 8	# 9	Total
Score										
Max	13	13	9	8	12	8	7	10	10	90

1. (*13 points*) ..... Short Answer

(a) (*2 points*) Suppose a remote attacker is able to learn the value of the random canary used by StackGuard (e.g., due to a format string bug). Can the attacker circumvent the StackGuard buffer overflow protection mechanism? If so, explain how. If not, explain why not.

(b) (*2 points*) What is a "time of check, time of use" bug?

(c) (*6 points*) In the DoS lecture we discussed a defense mechanism based on client puzzles.

a. How do client puzzles help in mitigating DoS attacks? (it suffices to focus on application-layer attacks such as an HTTP request flood).

b. Some puzzles are CPU-bound while others are memory-bound. What is the benefit of memory-bound puzzles over CPU-bound puzzles?

(d) (*3 points*) How does IPsec defend against packet replay attacks?

2. (*13 points*) ..... Worm detection

The Internet Storm Center (ISC) has a number of honeypots at a few fixed (and secret) IP addresses on the Internet, say n IP addresses total. The ISC puts out daily reports of worm activity. To be concrete, suppose the ISC publishes daily counts of the total number of infection attempts on port 137 (netbios) on all n sensors in the last 24 hours. You may assume that on a quiet day each sensor sees less than 50 attempts.

(a) (*2 points*) Suppose a worm developer knows the location of all n ISC sensors. Explain how he/she could write a fast spreading worm that is not be detectable by the ISC.

(b) (*6 points*) Consider an attacker who controls a large bot army. Describe an algorithm that enables the attacker to locate one of the ISC sensors within 32 days. You may assume that the bot army can send $100 \cdot 2^{31}$ packets within one day.
Hint: "32" refers to the number of bits in an IP address.

(c) (*5 points*) How long will it take to discover all n sensors? Give the best algorithm you can.

3. (*9 points*) ..... Distributed denial of service

Suppose that a web site uses a service like prolexic to defend itself against packet floods. The site comes under an attack from a bot army that generates a flood of normal-looking HTTP requests.

(a) (*2 points*) Can a packet flood filter (like prolexic) stop the attack?

(b) (*3 points*) Suppose the site maintains a list of past IP addresses that legitimately browsed the site before the attack started. Describe an architecture that would enable the site to keep servicing those machines on the list.

(c) (*4 points*) Can you suggest a way that the bot-net can adapt to defeat your defense measure from part (b)?

4. (*8 points*) ..... Hashes and MACs

A Message Authentication Code (MAC) consists of two algorithms (S, V). Algorithm $S(k, m)$ uses a secret key k to generate an integrity tag for a message m. Algorithm $V(k, m, t)$ uses a secret key k to validate a given integrity tag t for m.

Recall that a cryptographic hash function h is a non-keyed function that outputs a short hash $h(m)$ for an input message m. The function is said to be collision resistant if it is difficult to find a collision: two distinct messages m_0, m_1 such that $h(m_0) = h(m_1)$.

Let us consider four mechanisms for providing file integrity for a single file F on disk. The file system must be able to detect any unauthorized modification to this file. We say that the system is secure if an attacker cannot modify F without being detected. You may assume that the owner of file F has a password known to the system, but not to the attacker.

- Method 1: Compute an integrity tag for file F and store the integrity tag in the header of F. Upon file open the file system checks that the integrity tag is valid.

  (a) (*2 points*) Suppose the integrity tag is computed using a collision resistant hash function applied to F. Validating the integrity tag upon file open is done by rehashing the file and comparing the result to the value in the file header. Is the resulting system secure?

  (b) (*2 points*) Suppose the integrity tag is computed as the MAC of F using the user's password as the MAC secret key. Is the resulting system secure?

- Method 2: Compute an integrity tag for file F and store the integrity tag in read only memory (say, a disk partition that the attacker can read but not modify).

  (a) (*2 points*) Suppose the integrity tag is computed using a collision resistant hash function. Is the resulting system secure?

  (b) (*2 points*) Suppose the integrity tag is computed using a MAC with the user's password as the secret key. Is the resulting system secure?

In all sections, please justify your answer.

5. (*12 points*) ..... Software fault isolation

This problem asks about software-based fault isolation. To make the question more concrete, we will examine the techniques as applied to DEC alpha assembly language, in which all instructions are 32-bits wide and must start on an aligned 4-byte boundary. We will make use of the following assembly language instructions:

- `br` *Ra*`,`*disp*
  An unconditional branch, relative to the next instruction. This instruction stores the address of the next instruction in *Ra* and then skips *disp* instructions (where *disp* may be negative). For example `br r31,+6` skips 6 instructions, while `br r31,-10` branches back 10 instructions (as is done in a loop).
- `jmp` *Ra*`,(`*Rb*`)`
  Jump to register. Stores the address of the next instruction in *Ra*, then jumps and starts executing code at address *Rb*.
- `ldq` *Rv*`,`*disp*`(`*Ra*`)`
  Load instruction. Register *Ra* is a register containing an address in memory. *disp* is a 16-bit signed number. This instruction loads the value of memory location *Ra*+*dist* into register *Rv*.
- `stq` *Rv*`,`*disp*`(`*Ra*`)`
  Store instruction. Register *Ra* is a register containing an address in memory. *disp* is a 16-bit signed number. This instruction stores the value of register *Rv* into memory location *Ra*+*dist*.
- `bis` *Ra*`,`*Rb*`,`*Rc*
  Logical or instruction, named bis for "bit set." Sets register *Rc* to the bitwise or of *Ra* and *Rb*. (The equivalent of the C or java code "`Rc = Ra | Rb`;")
- `and` *Ra*`,`*Rb*`,`*Rc*
  Logical and instruction. Sets register *Rc* to the bitwise and of *Ra* and *Rb*. (The equivalent of the C or java code "`Rc = Ra & Rb`;")

(a) (*2 points*)

Fault-isolation requires inserting special checking code before every *unsafe instruction*. An example of an unsafe instruction is a store, `stq` ***Ra***`,0(`***Rb***`)`, when the address in ***Rb*** cannot be statically verified to lie within a fault domain's data segment. (Statically verified means verified by examining the code before it is actually executed.) Which of the following other instructions can be unsafe?

Circle the best answer; only one is correct:

i. `br` ***Ra***`,`***disp*** (where ***disp*** falls within the fault domain's code segment)

ii. `jmp` ***Ra***`,(`***Rb***`)`

iii. `ldq` ***Rv***`,`***disp***`(`***Ra***`)`

iv. `bis` ***Ra***`,`***Rb***`,`***Rc***

v. `and` ***Ra***`,`***Rb***`,`***Rc***

(b) (*5 points*)

One technique to control unsafe store instructions is *sandboxing*. Consider storing the value of register `r2` to the memory address in register `r1`. In unsafe code, this would be the single instruction:

```
stq r2, 0(r1)
```

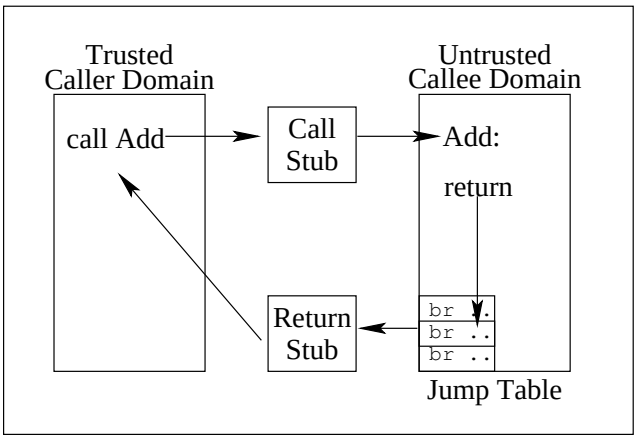
A sandboxed store requires three dedicated registers—`dedicated-reg`, `and-mask-reg`, and `segment-reg`, for which we will use registers `r25`, `r24`, and `r23` respectively. Here is the sandboxed equivalent of the above store instruction:

```
and r1, r24, r25      ; dedicated-reg ⇐ target-reg & and-mask-reg
bis r25, r23, r25     ; dedicated-reg ⇐ dedicated-reg | segment-reg
stq r2, 0(r25)        ; store instruction uses dedicated-reg
```

Other than these stylized store sequences, the checker rejects any code that modifies dedicated registers `r23`–`r25`. Suppose, however, that we did away with `r25` and instead sandboxed store instructions as follows (`r1` is not dedicated):

```
and r1, r24, r1
bis r1, r23, r1
stq r2, 0(r1)
```

Describe a way to subvert the safety of a system that uses this modified sequence.



Trusted code making an RPC into an untrusted fault domain to call the "Add" function.

(c) (*5 points*)

The above figure (Figure 4 from the SFI paper, which we also saw in lecture) illustrates how cross-fault-domain communication is implemented. A small region of a fault domain's code segment, the *jump table*, consists of a series of `br` instructions, each of which branches off to a small *stub* sequence outside of the code segment. The stub is trusted code that copies arguments and fixes up machine state such as registers that must be changed when switching fault domains.

The jump table is the only part of an untrusted fault domain's code segment that can have instructions branching outside of that code segment. Thus, code in other parts of the segment cannot branch directly to stubs.

Suppose we did away with jump tables by instead embedding trusted stub sequences directly into a special "stub region" of the code segment. (This region could contain unsafe instructions, because the stubs are supplied by the trusted execution environment.) Untrusted code could then branch directly to the trusted stubs.

Explain how malicious code can subvert the safety of this proposed jump-table-free scheme.

6. (*8 points*) ..... TCP spoofing

Simple Syndication Ads (ssads) is an ad syndication service that publishers can use to run advertisements on their web sites. Ssads generates banners that link to URLs of the following form:

```
http://ssads.com/pub=3942&adv=964&landing=http://www.yoursite.com/
```

Here `3942` is the publisher's account number, `964` is the advertiser's account number, and `www.yoursite.com` is the web site to which the advertiser wishes to send people who click on the banner. When `ssads.com` receives a request for such a URL, it charges the advertiser's account for a click, credits the publisher's account for a click, and generates an HTTP redirect to `www.yoursite.com`.

(a) (*3 points*)

Suppose that an attacker breaks the pseudo-random generator used by `ssads.com`'s TCP implementation for initial sequence numbers, and thus can guess the server's TCP sequence numbers with relatively good probability. Explain how a malicious publisher could exploit this fact to perpetrate click fraud.

(b) (*5 points*)

Give one technique that `ssads.com` could use to thwart such attacks, even if its TCP sequence numbers are easily guessed.

7. (*7 points*) ..... Watermarking

Watermarking is used to embed signals in music and video. A watermarking system consists of two algorithms Embed and Retrieve.

- Embed(k, m, d) embeds data d into an object (e.g. music file) m using secret key k. It outputs a new object $\hat{m}$.
- Retrieve$(k, \hat{m})$ outputs the watermark embedded in $\hat{m}$.

Clearly Retrieve$(k,$ Embed$(k, m, d)\ ) = d$. Moreover, the watermarking system must satisfy the following two informal properties:

a The output of Embed is of the same quality as the input m (as applies to audio or video data).

b Let $\hat{m}$ be the output of Embed(k, m, d). Then an attacker who is given $\hat{m}$ cannot create a new *usable* object m' such that Retrieve$(k, m') \neq d$.

Suppose a web site such as National Geographic (NG) wishes to use watermarking to find web sites who copied images from the NG site onto their own site.

(a) (*2 points*) Explain how National Geographic would go about locating infringing sites? What would go wrong if property (b) did not hold?

(b) (*5 points*) Suppose an NG subscriber posts an NG image anonymously to a public web site. NS wishes to identify the subscriber so that his/her access rights to the NG site can be revoked. To do so, for each image download from the NG site, NG embeds the subscriber's 32-bit IP address as the watermark data d in the image — NG embeds the first 16 bits in the top half of the image and embeds the last 16 bits in the bottom half of the image.

Show that this proposal is completely insecure (without resorting to network level attacks): an attacker can create images that cannot be traced back to the attacker. **Hint:** The attacker has access to multiple IP addresses (but NG knows that all of them belong to the attacker).

8. (*10 points*) ..... Anonymous browsing

(a) (*5 points*) The Tor system is designed to conceal the browser's IP address from the server. How would you design a Tor-like mechanism for the reverse property — allowing a web server to offer a service without revealing any information about the IP address of the server. Browser privacy is not important. As in standard Tor, server anonymity should be assured as long as one mix node in the chain is trustworthy. Design your system so that it requires no browser changes. Server changes are allowed.

(b) (*2 points*) Suppose a US client uses Tor to issue anonymous Google search queries. If the last hop in the Tor network happens to be in Germany, Google thinks the request is from a German user and answers in German with responses relevant to a German locality. Can you suggest a way to fix this problem?

(c) (*3 points*) Practical anonymity networks such as Tor use stateful mix nodes where nodes maintain per flow state. Explain why a stateful mix net is preferable to a stateless network.

9. (*10 points*) ..... TCG

In class we discussed the Trusted Computing architecture (TCG) and the associated TPM hardware.

(a) (*2 points*) Can the TPM be used to prevent a virus from modifying the machine's Master Boot Record (MBR), used for bootstrapping the OS, without being detected? If so, explain why. If not, explain why not.

(b) (*2 points*) Can the TPM be used to prevent a virus from modifying the machine's BIOS boot block without being detected? If so, explain why. If not, explain why not.

(c) (*2 points*) Suppose user A is able to extract the secret AIK signing key from the tamper resistant chip in his machine. Explain the implications of this for the validity of the attestation process. How could A use this key to fool a remote server about the software running on A's machine?

(d) (*4 points*) How would you defend against this problem? You may assume that the private key extracted from the chip is published on the web (anonymously) so that anyone can mount the attack from part (c).