I Web security

Ben Bitdiddle runs a popular web site at `bensblog.com`. Ben decides he wants to make some money on the side, and signs up with a third-party ad network. The ad provider gives him the following HTML snippet ("snippet 0"), and tells him to add it to his page where he wants ads to be shown.

```
<script src="http://adsadsads.com/ads.js?customer=bensblog">
</script>
```

Ben has taken 6.858 and decides to use HTTPS instead of HTTP to fetch the script ("snippet 1"):

```
<script src="https://adsadsads.com/ads.js?customer=bensblog">
</script>
```

Ben's site is also served over HTTPS. Both sites have valid and trusted TLS certificates, and their private keys have not been stolen.

1. [3 points]: Describe one attack that is possible with snippet 0 that is prevented by snippet 1? (Briefly explain your answer.)

Answer: A man in the middle of attack is possible with snippet 0. With snippet 1, an attacker cannot replace the message containing the javascript for ads.js with a message containing the attacker's javascript.

Ben reads about the `integrity` attribute. This is part of a security mechanism called *Subresource Integrity (SRI)* that is currently being standardized. An `integrity` attribute can be added to a `script` tag to indicate the expected cryptographic hash for the script being referenced. If the contents of the script that the browser downloads do not match the given hash, the browser will refuse to execute the script. The intent is that an SRI integrity check should be added to any script loaded from a third-party source to ensure that it has the expected content.

Ben adds an `integrity=` attribute to snippet 1 with the hash of the current script, so that his web pages instead include this snippet ("snippet 2"):

```
<script src="https://adsadsads.com/ads.js?customer=bensblog"
  integrity="sha384-R4/ztc4ZlRqWjqI...ZETq72KgDVJCp2TC">
</script>
```

2. [3 points]: Describe an attack that snippet 1 allows, but snippet 2 prevents. (Briefly explain your answer.)

Answer: With snippet 1, the ad network could start serving malicious code, either because they have been compromised, or because they are evil, which will again compromise Ben's site. With snippet 2, if Ben can include a hash of the ad network's ad script when it is in a non-malicious state, the attack against snippet 1 is prevented.

Ben is concerned with the loading time of his site, and opens up his browser's network request inspector to see how he might speed things up. He sees that the TLS handshake with the ad provider is taking several milliseconds. Ben figures that it is sufficient for his site alone to be served over HTTPS, and decides to further change the ad code to use HTTP instead of HTTPS to avoid the overhead of an additional TLS connection. Ben's ad code now looks like this ("snippet 3"):

```
<script src="http://adsadsads.com/ads.js?customer=bensblog"
  integrity="sha384-R4/ztc4ZlRqWjqI...ZETq72KgDVJCp2TC">
</script>
```

3. [3 points]: Is snippet 3 more secure, less secure, or equivalent in security to snippet 2? (Briefly explain your answer.)

Answer: The same.

Ben modifies his web site to have a login form. After a successful login, the site stores a cookie with the user's credentials in the user's browser. Ben runs this plan by Alyssa and she recommends that he arrange with the ad network to be able to put an iframe around snippet 1 as follows ("snippet 4"):

```
<iframe src="https://adsadsads.com/ads.html?customer=bensblog">
</iframe>
```

`https://adsadsads.com/ads.html?customer=bensblog` contains:

```
<script src="https://adsadsads.com/ads.js?customer=bensblog">
</script>
```

4. [3 points]: Give an example of an attack that is possible without the iframe (snippet 1) but impossible with the iframe (snippet 4).

Answer: The javascript in snippet 1 cannot steal a user's cookie if snippet 0 is in a iframe, because the SOP doesn't allow code inside the iframe to refer to resources outside of the iframe.

II Ur/Web

Ben is excited about Ur/Web, as described in the paper "Ur/Web: A simple model for programming the web" by Chlipala. He rewrites the `zoobar` application from Lab 4 using Ur/Web.

5. [3 points]: Identify a specific security exploit from Lab 4 that would be eliminated by using Ur/Web. Briefly explain why Ur/Web would eliminate that attack.

Answer: Many right answers. For example, code injection attacks. In Ur/Web objects have an appropriate, unchangeable type. It doesn't allow string fragments to be interpreted as code objects, which is is the basic of reflected cross-site scripting attack in exercise 3 of lab 4.

III SYN Floods

Recall the TCP discussion in Lecture 12: before 1996, server TCP implementations would handle arrival of a SYN (connection setup) packet by remembering the connection, and by replying with a SYN+ACK packet containing the server's initial sequence number (ISN). The server expects an ACK packet from the client echoing the server's ISN. The server would keep state for this "half-open" connection for many minutes. To avoid running out of memory, server TCP implementations would limit the number of half-open connections allowed to exist; if a SYN arrived when too many already existed, the server would ignore the SYN (and thus not allow a connection to be set up).

In the "SYN flood" denial-of-service attack described in the lecture the attacker sends TCP SYN packets to a server at a low rate, perhaps a few per second. These SYN packets have random source IP addresses — that is, the source is forged. The attacker would not send the ACK packets the server expected. The result is that the server almost always had the maximum number of half-open connections, and thus ignored legitimate client connection requests.

A charming (for the attacker) feature of this attack is that it required only a very low rate of attack packets, so it was easy to mount the attack from a single computer with a cheap Internet access link. Further, the low volume would likely not raise any alerts from network monitoring systems looking for large volumes of traffic.

6. [3 points]: Suppose Bob is aiming a SYN flood attack from his home computer against www.anti-bob-club.com. Why can't the server at www.anti-bob-club.com recognize the packets from Bob's computer and ignore them?

Answer: The packets look just like ordinary connection setup requests from ordinary clients. They do not contain Bob's IP address.

The "SYN cookie" defense described in Lecture 12 defeats SYN flood attacks. The idea is that the server's TCP responds to a SYN packet with a SYN+ACK packet containing the following ISN (some details omitted):

```
ISN = SHA1(source IP address, secret)
```

The secret is a value that only the server knows; the source IP address is the source address in the SYN packet. The server does *not* keep any state about this half-open connection, and thus does not limit the number of half-open connections that can exist. If an ACK arrives at the server for a connection that doesn't exist, the server checks whether the sequence number being acknowledged is equal to the ISN computed above. If it is, the server concludes that the ACK is part of a legitimate connection setup sequence, and the server creates state for a new connection. Otherwise the server ignores the ACK packet.

www.anti-bob-club.com uses SYN cookies.

7. [3 points]: Bob thinks maybe he can just wait for the server's SYN+ACK packet to arrive, look at the ISN it contains, and send that ISN back to the server in an ACK. That would cause the server to allocate memory for a new connection; if Bob created enough connections like this, and never closed them, perhaps he could force the server to run out of memory. Why won't Bob's idea work?

Answer: If Bob is using random IP source addresses, he won't see the SYN+ACK packets, so he can't send the ISNs back to the server. If Bob is using his own source IP address, then the server's administrators will fix their firewall to drop his packets.

8. [3 points]: Explain why the secret is an important part of the server's ISN computation.

Answer: If there were no secret, the attack could easily generate the correct ISN for any source IP address. The secret forces the attacker to do billions of SHA1 computations in order to come up with an acceptable ISN; it's probably best if the server changes the secret that often.

IV Kerberos

Alice, an MIT student, is designing a secure lab submission system. The goal is to let students in various courses submit their lab solutions for grading. Alice's design gives each course its own lab submission server. It's important that students upload their labs to the correct server for their course, to avoid unwanted disclosure of their solutions. For example, if rogue students were to set up fake submission servers, Alice's system should not accidentally upload solutions to those servers.

Alice wants to use Kerberos to provide security (see *Kerberos: An Authentication Service for Open Network Systems*). She gives each course's server its own Kerberos principal. For example, 6.858's server has Kerberos principal "UP858," the Kerberos KDC has an entry for "UP858" with a corresponding key, and 6.858's server knows that key. Alice writes an application that students can run on Athena workstations that uses Kerberos to establish a secure channel to a submission server. Assume that the cryptography underlying Kerberos secure channels ensures confidentiality and integrity.

Students need a way to tell Alice's application the correct server to which to upload their lab. Alice thinks it would be great if instructors could write the information students need on the blackboard on the first day of class.

9. [3 points]: Can Alice design the system in this way? If yes, what should instructors write on the board, and why does that work? If no, why not?

Answer: Yes. The instructor should write the name of the Kerberos principal for the class's upload server, e.g. "UP858". This works because Alice's application can ask Kerberos TGS for a ticket to UP858, which only the real UP858 server can decode to find the secure channel keys to talk to Alice.

Suppose the MIT AFS server is down, but Bob doesn't know that. Bob's workstation uses Kerberos to try to set up a connection to the AFS server. Eve sets up a fake server that sends and receives packets using the AFS server's IP address; the LAN sends the packets Bob addresses to the AFS server to Eve's server instead. Eve runs her own software on her server, which mimics the real MIT AFS server as best it can, though it does not know the real AFS server's Kerberos key. Eve cannot compromise the KDC.

Please answer the next two questions based on the Kerberos protocol as described in the paper (Section 4) and/or the notes for Lecture 13. (If you are familiar with the real Kerberos protocol implementation, you may realize that some of its details could affect the answers; but please answer with respect to the paper's description.)

10. [3 points]: At what point in the Kerberos protocol (if any) will Bob's workstation realize that there is a problem?

Answer: Eve's server won't be able to produce the encrypted TS+1 message in step 6 of the protocol in the notes or the paper's Figure 7, because Eve's server does not know the session key. So Bob's workstation will either never receive that message, or it will receive a message that decrypts to the wrong TS.

Now suppose that Eve can also modify Bob's packets on the LAN. Eve modifies Bob's message to the TGS server (Figure 8 in the paper, or step 3 in the notes) to mention "eve@mit.edu" instead of the Kerberos ID of the AFS server; note that the server name in this message is not covered by encryption. Eve *does* know the Kerberos password for eve@mit.edu.

11. [3 points]: At what point in the Kerberos protocol (if any) will Bob's workstation realize that there is a problem?

Answer: The Kerberos protocol won't reveal the problem to Bob, since Eve's bad server will be able to decrypt and encrypt with the session key that Bob received from the TGS.

V TLS and HTTPS

12. [7 points]: Assume that a TLS connection has been established successfully between a client and a server. Establishing the session included checking the server certificate and executing a Diffie-Helmann exchange, but the client did not provide a client certificate. Further, assume that the client and server are honest, that the client and server don't leak their keys, and that the cryptography is good. Which of the following attacks does TLS protect against?

(Circle True or False for each choice.)

A. True / False An attacker replacing bytes sent by a client with bytes of the attacker's own choosing.

Answer: True.

B. True / False An attacker reading the plaintext bytes sent by a client.

Answer: True.

C. True / False An attacker replaying bytes that a client sent earlier.

Answer: True.

D. True / False An attacker impersonating the server.

Answer: True.

E. True / False An attacker impersonating the client.

Answer: True/False. The client isn't authenticated but remains consistent for the duration of the session.

F. True / False An attacker stealing the server's private key and reading the plaintext of recorded past connections.

Answer: True.

G. True / False An attacker breaking into a certificate authority and creating a fake certificate for the server.

Answer: False.

Consider a browser that retrieves a web page over HTTPS. A Web developer by accident incorporated a script in that page as follows:

```
<script src="http://jquery.com/jquery1.4.1.min.js"></script>
```

13. [3 points]: What does ForceHTTPS (as described in the paper by Jackson and Barth) do to avoid such accidents? (Explain your answer briefly.)

Answer: It allows the server administrator to set a ForceHTTPS cookie, which forbids mixed http and https content. Refer to section 1.1, bullet 3: "Attempts to embed insecure (non-HTTPS) content into the site fail with network errors."

14. [3 points]: Ben modifies FireFox to turn every certificate error into "a page not found" error message. He reasons this improves security because it prevents users from clicking through invalid certificates. Many users download his modified browser, but quickly stop using it. Why would users want to give up on the better security offered by Ben's browser?

Answer: Certificates are often incorrect accidentally. For example, an administrator forgot to renew the certificate for his/her web site in time. Using Ben's browser would mean that many Web sites are inaccessible. Users care more about usability than security.

VI Side channels

15. [6 points]: The openSSL implementation described in "Remote Timing Attacks are Practical" (by Brumley and Boneh) uses the following performance optimizations: Chinese Remainder (CR), Montgomery Representation (MR), Karatsuba Multiplication (KM), and Repeated squaring and Sliding windows (RS). Which of the following options would close the timing channel attack described in the paper if you turned the listed optimizations **off**?

(Circle True or False for each choice.)

A. True / False CR, MR, KM, and RS.

Answer: True

B. True / False RS

Answer: False

C. True / False RS and KM

Answer: False

D. True / False RS and MR

Answer: True

E. True / False CR and MR

Answer: True

F. True / False CR

Answer: True

Answer: Without CR, you do things modulo N, not q and learning N isn't important. Without MR you don't see the extra reduction, which is the main source of the timing channel.

16. [3 points]: Ben launches the remote timing attack described in the paper from MIT on a server running at Stanford. He produces a graph in the style of Figure 3a of the paper, but he is unable to guess q. Why is that the case? How might his graph look like? (Briefly explain your answer.)

Answer: Internet rounds trips are in the order of 10s of msec and he needs to measure 10s of microseconds, which will be dwarfed by the variation in round-trip times. His graph might oscillate around the zero line: one bit above, next bit below, etc.

VII Tor

Fred wants to make a complaint about 6.858 to the staff, but he doesn't want the staff to know the complaint is from him. He connects from a PC in his dorm room through Tor to the 6.858 complaint-submission web server and submits some text. By a cruel twist of fate, the first Onion Router (OR) in the multi-OR Tor circuit he sets up is controlled by the 6.858 staff. Thus the 6.858 staff can observe that Fred is creating a Tor circuit, since it comes from Fred's IP address. The 6.858 staff have no special powers with respect to Tor other than this.

17. [3 points]: Can the first OR in Fred's circuit tell where Fred is connecting to? How, or why not?

Answer: It can't immediately tell, since Fred's OP encrypts that information so that only the exit OR can read it, and the first OR only knows Fred and the second OR's identities.

18. [3 points]: Are the 6.858 staff likely to be able to tell that the complaint submitted to their server is really from Fred, despite his use of Tor? Why or why not?

Answer: Probably. They know the time at which Fred set up a circuit, and the time at which a connection arrived from a Tor exit node to the 6.858 complaint server. If those times are close, they can guess that Fred may have submitted the complaint.

Alice uses Google Mail (gmail) through Tor. Bob claims that her use of Tor is pointless, since Alice has to log into gmail, so she can't be anonymous to gmail regardless of Tor.

19. [3 points]: Explain how Alice might benefit from using Tor despite Bob's objection.

Answer: Using Tor, she can hide her physical location (IP address) from gmail, and also prevent an external observer from realizing that she is talking to gmail.

Alice's gmail account may also have a name other than Alice, and she doesn't want gmail or an external observer to realize that she is the person behind that account.

20. [3 points]: Butler Lampson explained there is an opposition between security and convenience, and that convenience in general wins. Can you give an example in the design of Tor where convenience trumped security? (Briefly explain your answer.)

Answer: Tor doesn't delay traffic to mix it with other traffic because that would make Tor inconvenient use. But, having no cover traffic reduces Tor's anonymity guarantees.

VIII 6.858

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

21. [1 points]: Lab 5 was a new lab. On a scale of 1 (bad) to 10 (excellent) how would you rate the lab? What improvements would you recommend?

Answer:

29 more tests

17 code documentation

12 recitation on SUNDR

10 more, smaller exercises

10 clearer expectations (what to implement/how we grade)

22. [1 points]: Are there any papers in the second part of the semester that you think we should definitely remove next year? If not, feel free to say that.

Answer: ur/web (x42), Timing/Side Channel (x16), Tangled Web (x13), TCP/IP (x10), forcehttps (x8), TOR (x2), Haven (x2), OWASP (x2), kerberos (x2), Passwords

End of Quiz