April 25, 2016

Instructions. We will break into groups to discuss the following questions. Please think of as many solutions as you can. Be original! Maybe you will come up with something no one has thought of yet. Be prepared to talk about your solutions with the rest of the section.

Question 1 Videogames Cheating

(a) Suppose the client of a online game calculates the player’s current position locally and sends it back to the server as a pair (x, y). What can go wrong?

(b) How can you avoid this sort of problem?
Question 2  

Certificate Transparency  

One problem with current certificate management structure is that it lacks transparency. We don’t know what certificates are issued by the CAs, so it is hard to spot a problematic certificate.

(a) Suppose Mallory compromises a certificate authority, so he can issue any certificate that he want (Or maybe the certificate authority is just lazy). What can he do to pretend to be google.com?

(b) What are some ways to mitigate this problem? (Hint: We want to make it easy for people to know exactly what certificates have been published, so we should use somthing like a log here.)

(c) The solution you thought of above is actually called Certificate Transparency. It relies on certificate logs. What properties should the logs have in order to be secure?

(d) How can we construct the certificate logs to have these properties? (Hint: Think about project 2.)
Question 3  

Detecting packers  

(25 min)  

Most antivirus softwares today are still heavily signature-based. This means that they can only detect malware with known malicious code. To make malware harder to detect, malware authors often “pack” their program. This question will help you understand what “packers” are as well as some ways to detect them.

(a) Given a database of existing malicious code, what is the simplest way of detecting malware?

(b) Suppose Company Unsafe releases their new antivirus software Insecure. Insecure adopts the naive approach is the previous part. Mallory immediately realizes that he can easily make any piece of malware undetectable by Insecure. What is Mallory thinking of? (Hint: Think about buffer overflow).

(c) From the above example, we can see that packed programs usually have write-then-execute sequences. If we can detect this sequence, we can potentially detect the newly generated code inside memory. After taking CS161, Alice decides that she can detect this sort of packers herself. She decides to do it by page management. The malware expects the memory to be both writable and executable. However, in her system, when an executable runs, it would mark the executable region as read-only and executable, and everything else as read-only and non-executable. She also swapped out the page fault handler in her system to her custom program. How can she use this setup to detect write-then-execute sequences without damaging the functionality of normal programs?