Tentative Network Security Lab Project Topics

2017.2.23

Category	Topic	Description
Cryptography	**CAPTCHA cracking**	Step 1: Implement a fast CAPTCHA cracker based on deep learning. Step 2: Implement a CAPTCHA generator to generate enough training data. Step 3: In class, demonstrate that you can quickly recognize CAPTCHA. Explain the precision and recall of your approach. References • Multi-digit Number Recognition from Street View Imagery using Deep CNN • CAPTCHA Recognition with Active Deep Learning • http://matthewearl.github.io/2016/05/06/cnn-anpr/
IP security	**DNS cache poisoning**	Step 1: Implement Dan Kaminsky's fast DNS poisoning attack (another description), where the attacker brute forces responses to insert a false IP address into a DNS cache. Step 2: Demonstrate the countermeasure described in this paper. Step 3: In class, demonstrate the attack and how the implementation of the countermeasure succeeds to defend against the attack.
SSL/TLS	**Heartbleed**	Step 1: Implement the Heartbleed attack against OpenSSL. Step 2: Explain how the Heartbleed vulnerability is fixed, how we can defend against such attacks and why the Heartbleed vulnerability was missed for so long. Step 3: In class, demonstrate the attack and explain the vulnerability and its defense.
	TLS information leakage	Step 1: Implement the BEAST, CRIME, or Lucky 13 attack against TLS. Step 2: Explain how modern implementations attempt to defend against such side channels, and what the limitations of these defenses are.

	Fooling web users	Build a simple mock bank site and demonstrate: (1) an SSL stripping attack, where a man-in-the-middle transparently proxies HTTP requests and rewrites HTTPS links to point to look-alike HTTP links; (2) a clickjacking attack, as described here; (3) picture-in-picture and homograph attacks; (4) enhance your SSL attack to demonstrate the null prefix vulnerability (I'll provide a cert) and to defeat OCSP revocation.
WiFi	**Inferring smartphone password via WiFi signals**	Step 1: Implement the WindTalker attack as describe here. Step 2: In class, demonstrate the attack, and explain how modern implementations attempt to defend against such side channels. Reference: When CSI Meets Public WiFi: Inferring Your Mobile Phone Password via WiFi Signals
Web security	**Web tracking**	Step 1: Implement and demonstrate CSS history sniffing and timing-based history sniffing. Step 2: Select a widely deployed web bug and instrument your browser to monitor it. Display the data the web bug reports as it tracks you across various popular sites. Step 3: Explain how sites could use client-side tracking to show targeted ads with far less invasion of privacy.
Cloud security	**Container image security**	Step 1: Understand the detailed design of static vulnerability analyzer Clair. Step 2: Setup the environment for the static vulnerability analyzer Clair. Step 3: Analyze multiple container images from a public registry, and explain the findings. Step 4: In class, explain the design of Clair, demonstrate how Clair could be used to detect vulnerabilities, and explain the findings. Reference: https://github.com/coreos/clair