RUHR-UNIVERSITÄT BOCHUM

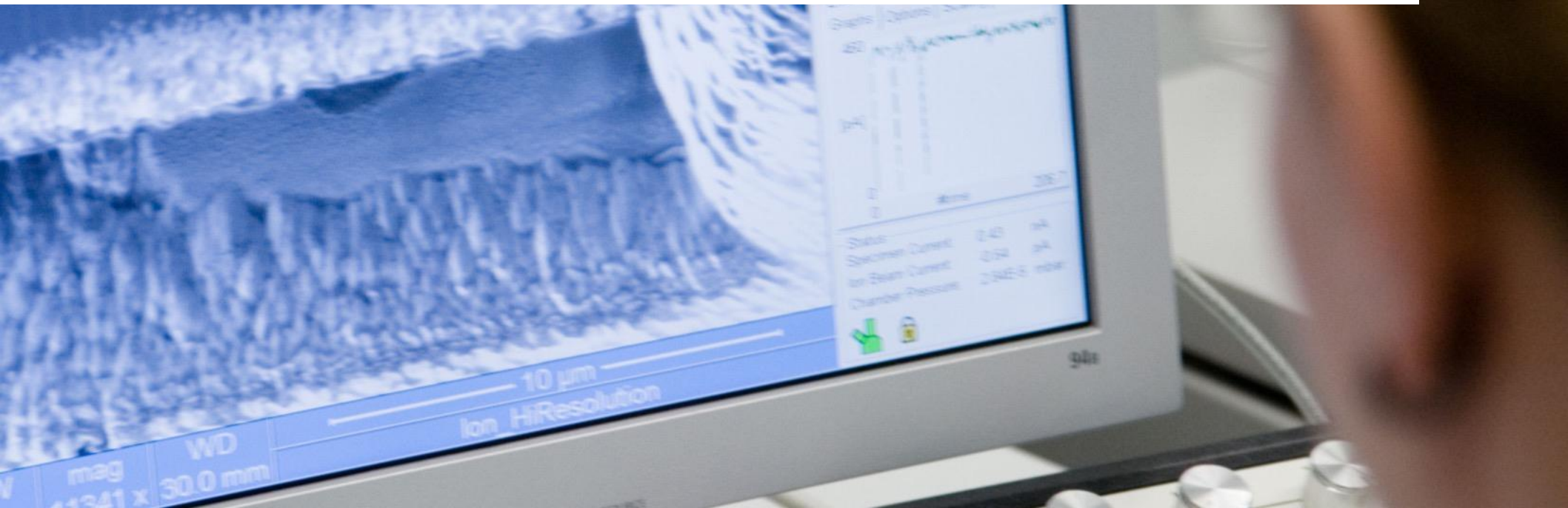
RUB

Kickoff Meeting

Modul Proseminar

18.04.2016

hgi Lehrstuhl für Netz- und Datensicherheit



Agenda

- Vorstellung
- Zeitplan/ Organisation

Vorstellung

- M.Sc. Sebastian Lauer
  - Wissenschaftlicher Mitarbeiter am Lehrstuhl für Netz- und Datensicherheit der Ruhr-Uni Bochum.
  - Modulentwickler im Projekt OpenC^{3}S.
  - Forschungsschwerpunkt: Grundlagen der Public Key Kryptographie.
  - Authentische Schlüsselvereinbarung
  - Sebastian.lauer@rub.de

Vorstellung

- M.Sc. Christopher Späth
  - Wissenschaftlicher Mitarbeiter am Lehrstuhl für Netz- und Datensicherheit der Ruhr-Uni Bochum.
  - Modulentwickler im Projekt OpenC^{3}S.
  - Forschungsschwerpunkt: Web-Security und XML
  - Christopher.spaeth@rub.de

Zeitplan/ Organisation

1. KickOff 18.04.2016

2. Bis 02.05.Themenauswahl: Bitte schickt an eure Betreuer das Thema, welches ihr bearbeiten wollt, und ein Alternativthema (um Überschneidungen zu vermeiden). **Eigene Themen sind auch erlaubt**

3. Bis 06.06. Expose (dazu gleich mehr)

4. Bis 29.07: Vorabversion

5. Bis 02.09. Endversion

6. Präsentation am 24.09. in Bochum

Exposé

- 2-3 DIN A4 Seiten
- Geplante Inhalte darstellen
  - Szenario
  - Fragestellung
  - Vorgehensweise
  - Literatur
  - Kriterien für Erfolg

Wieso ein Exposé?

Vorwissen vorhanden?

Voraussetzungen erfüllt?

Einarbeitungszeit notwendig?

Text aus Expose = Einführung der Seminararbeit

Lieber ein gutes Expose als ...

Seminararbeit

- Ca. 8-10 Seiten

- Mit Latex!

Mehr Infos und Vorlagen zu finden auf:

http://nds.rub.de/teaching/lectures/23/

http://nds.rub.de/teaching/theses/seminar/

Themen

1.Attacks Only Get Better: Password Recovery
Attacks Against RC4 in TLS

http://0b4af6cdc2f0c5998459-c0245c5c937c5dedcca3f1764ecc9b2f.r43.cf2.rackcdn.com/20966-sec15-paper-garman.pdf

2. Making Bitcoin Exchanges Transparent
http://www.tik.ee.ethz.ch/file/b89cb24ad2fa4e7ef01426d318c9b98b/decker2015making.pdf

3. Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice
https://weakdh.org/imperfect-forward-secrecy-ccs15.pdf

4. Face/Off: Preventing Privacy Leakage From Photos in Social Networks

Themen

Leave me alone: App-level protection against runtime information gathering on Android
http://www.ieee-security.org/TC/SP2015/papers-archived/6949a915.pdf
(S&P 2015)

Web-to-Application Injection Attacks on Android
http://www.comp.nus.edu.sg/~prateeks/papers/W2AI.pdf
(ESORICS 2015)

Identifying Cross-origin Resource status using Application Cache
http://www.internetsociety.org/sites/default/files/01_1_2.pdf
(NDSS 2015)

Themen

Password Managers: Attacks and Defenses
http://0b4af6cdc2f0c5998459-c0245c5c937c5dedcca3f1764ecc9b2f.r43.cf2.rackcdn.com/16381-sec14-paper-silver.pdf
(Usenix 2014)

Precise Client-Side Protection against DOM-based Cross-Site Scripting
http://0b4af6cdc2f0c5998459-c0245c5c937c5dedcca3f1764ecc9b2f.r43.cf2.rackcdn.com/16395-sec14-paper-stock.pdf
(Usenix 2014)

Leave me alone: App-level protection against runtime information gathering on Android

- http://www.ieee-security.org/TC/SP2015/papers-archived/6949a915.pdf (S&P 2015)

Abstract—Stealing of sensitive information from apps is always considered to be one of the most critical threats to Android security. Recent studies show that this can happen even to the apps without explicit implementation flaws, through exploiting some design weaknesses of the operating system, e.g., shared communication channels such as Bluetooth, and side channels such as memory and network-data usages. In all these attacks, a malicious app needs to run side-by-side with the target app (the victim) to collect its runtime information. Examples include recording phone conversations from the phone app, gathering WebMD's data usages to infer the disease condition the user looks at, etc. This *runtime-information-gathering* (RIG) threat is realistic and serious, as demonstrated by prior research and our new findings, which reveal that the malware monitoring popular Android-based home security systems can figure out when the house is empty and the user is not looking at surveillance cameras, and even turn off the alarm delivered to her phone.

To defend against this new category of attacks, we propose a novel technique that changes neither the operating system nor the target apps, and provides immediate protection as soon as an ordinary app (with only normal and dangerous permissions) is installed. This new approach, called *App Guardian*, thwarts a malicious app's runtime monitoring attempt by pausing all suspicious background processes when the target app (called *principal*) is running in the foreground, and resuming them after the app stops and its runtime environment is cleaned up. Our technique leverages a unique feature of Android, on which third-party apps running in the background are often considered to be disposable and can be stopped anytime with only a minor performance and utility implication. We further limit such an impact by only focusing on a small set of *suspicious* background apps, which are identified by their behaviors inferred from their side channels (e.g., thread names, CPU scheduling and kernel time). App Guardian is also carefully designed to choose the right moments to start and end the protection procedure, and effectively protect itself against malicious apps. Our experimental studies show that this new technique defeated all known RIG attacks, with small impacts on the utility of legitimate apps and the performance of the OS. Most importantly, the idea underlying our approach, including app-level protection, side-channel based defense and lightweight response, not only significantly raises the bar for the RIG attacks and the research on this subject but can also inspire the follow-up effort on new detection systems practically deployable in the fragmented Android ecosystem.

Web-to-Application Injection Attacks on Android

http://www.comp.nus.edu.sg/~prateeks/papers/W2AI.pdf
(ESORICS 2015)

Abstract. Vulnerable Android applications are traditionally exploited via malicious apps. In this paper, we study an underexplored class of Android attacks which do not require the user to install malicious apps, but merely to visit a malicious website in an Android browser. We call them web-to-app injection (or W2AI) attacks, and distinguish between different categories of W2AI side-effects. To estimate their prevalence, we present an automated W2AIScanner to find and confirm W2AI vulnerabilities. Analyzing real apps from the official Google Play store – we found 286 confirmed vulnerabilities in 134 distinct applications. Our findings suggest that these attacks are pervasive and developers do not adequately protect apps against them. Our tool employs a novel combination of static analysis and symbolic execution with dynamic testing. We show through experiments that this design significantly enhances the detection accuracy compared with an existing state-of-the-art analysis.

RUB

Identifying Cross-origin Resource status using Application Cache

http://www.internetsociety.org/sites/default/files/01_1_2.pdf
(NDSS 2015)

***Abstract*—HTML5 Application Cache (AppCache) allows web applications to cache their same- and cross-origin resources in the local storage of a web browser to enable offline access. However, cross-origin resource caching in AppCache has potential security and privacy problems. In this paper, we consider a novel web privacy attack that exploits cross-origin AppCache. Our attack allows a remote web attacker to exploit a victim web browser to exactly identify the status of target URLs: existence, redirection, or error. Especially, our attack can be performed without using client-side scripts, can concurrently identify the status of multiple URLs, and can exactly identify the redirections of target URLs. We further demonstrate advanced attacks that leverage the basic attack to de-anonymize and fingerprint victims. First, we determine the login status of a victim web browser by identifying URL redirections or errors due to absent or erroneous login information. Second, we probe internal web servers located in the local network of a victim web browser by identifying URL existence. We also suggest an effective countermeasure to mitigate the proposed attacks.**

RUB

Password managers: Attacks and Defenses

http://0b4af6cdc2f0c5998459-c0245c5c937c5dedcca3f1764ecc9b2f.r43.cf2.rackcdn.com/16381-sec14-paper-silver.pdf
(Usenix 2014)

Abstract

We study the security of popular password managers and their policies on automatically filling in Web passwords. We examine browser built-in password managers, mobile password managers, and 3rd party managers. We observe significant differences in autofill policies among password managers. Several autofill policies can lead to disastrous consequences where a remote network attacker can extract multiple passwords from the user's password manager without any interaction with the user. We experiment with these attacks and with techniques to enhance the security of password managers. We show that our enhancements can be adopted by existing managers.

Precise Client-Side Protection against DOM-based Cross-Site Scripting

http://0b4af6cdc2f0c5998459-c0245c5c937c5dedcca3f1764ecc9b2f.r43.cf2.rackcdn.com/16395-sec14-paper-stock.pdf
(Usenix 2014)

Abstract

The current generation of client-side Cross-Site Scripting filters rely on string comparison to detect request values that are reflected in the corresponding response's HTML. This coarse approximation of occurring data flows is incapable of reliably stopping attacks which leverage non-trivial injection contexts. To demonstrate this, we conduct a thorough analysis of the current state-of-the-art in browser-based XSS filtering and uncover a set of conceptual shortcomings, that allow efficient creation of filter evasions, especially in the case of DOM-based XSS. To validate our findings, we report on practical experiments using a set of 1,602 real-world vulnerabilities, achieving a rate of 73% successful filter bypasses. Motivated by our findings, we propose an alternative filter design for DOM-based XSS, that utilizes runtime taint tracking and taint-aware parsers to stop the parsing of attacker-controlled syntactic content. To examine the efficiency and feasibility of our approach, we present a practical implementation based on the open source browser Chromium. Our proposed approach has a low false positive rate and robustly protects against DOM-based XSS exploits.