ISSN 2186-7437

NII Shonan Meeting Report

No. 159

Web Application Security

Limin Jia
Tamara Rezk
Sukyoung Ryu

March 18–21, 2024

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

List of participants

Pedro Adão, Instituto Superior Técnico and Instituto de Telecomunicações, Portugal
Musard Balliu, KTH Royal Institute of Technology, Sweden
Davide Balzarotti, Eurecom, France
David Basin, ETH Zurich, Switzerland
Lujo Bauer, Carnegie Mellon University, USA
Abhishek Bichhawat, Indian Institute of Technology Gandhinagar, India
Luca Compagna, SAP Security Research, France
Steven Englehardt, DuckDuckGo, USA
Hsu-Chun Hsiao, National Taiwan UniversityTaiwan
Limin Jia, Carnegie Mellon University, USA
Christoph Kerschbaumer, Mozilla, Germany
Zhenkai Liang, National University of Singapore, Singapore
Aastha Mehta, University of British Columbia, Canada
John Mitchell, Stanford University, USA
Klaas Pruiksma, University of Stuttgart, Germany
Tamara Rezk, INRIA, France
Xavier Rival, INRIA,France
Sebastian Roth, TU Wien, Austria
Sukyoung Ryu, KAIST, South Korea
Merve Sahin, SAP Security Research, France
David Sands,Chalmers University of Technology, Sweden
Cristian-Alexandru Staicu, CISPA Helmholtz Center for Information Security, Germany
Dolière Francis Somé, CISPA Helmholtz Center for Information Security, Germany
Ben Stock, CISPA Helmholtz Center for Information Security, Germany
Tachio Terauchi, Waseda University, Japan
John Wilander, USA

Introduction and Meeting Overview

Today's web applications are a mix of existing online libraries and data that are combined to write applications rapidly and inexpensively. Moreover, the last decades have witnessed an accelerating trend to integrate not only documents and code but also the so-called Web of Things that uses web applications to connect homes, cars, appliances, and other physical devices. However, this same flexibility together with the mix of heterogeneous technologies makes the task of programming secure web applications and protecting users against exploits very complex. As web applications are becoming essential in people's lives, web and browser vulnerabilities as well as privacy issues associated with web technologies such as tracking and fingerprinting have become a major threat that people face today. Challenges regarding security and privacy issues of web technology include the handling of injections in clients and servers due to the mix of technologies, the inclusion of untrusted code as a common practice, the protection of web sessions implemented over HTTP, the lack of languages available on the client side, the complexity of the JavaScript language, the main language for web pages, and the complexity of the browser infrastructure. Developers and users are facing an unprecedented need for security mechanisms to help identify, mitigate, and remove web vulnerabilities.

This meeting provided a forum to

- Discuss recent developments and issues in security and privacy of web technology.
- Discuss the effectiveness of security mechanisms in face of the current overall vulnerability landscape

In particular, we addressed the following questions:

- What do formal methods bring to web security and privacy in practice?
- Which security analyses are appropriate in face of the heterogeneity of technologies required in modern web applications?
- The heterogeneity and new security threats of Web of Things technology makes analysis or enforcing security policies even more difficult: which are the new security and privacy concerns in the Web of Things?
- How to bring state of the art to practice? What are the actual obstacles that prevent the technology from being applied?
- The steep learning curve (usability of the tool), and infrastructure dependency makes it difficult to keep a tool up to date with the newest infrastructure (e.g., tools that require heavy modification of software infrastructure such as Chrome simply cannot keep up with Google's frequent updates to Chrome). How to develop techniques that are infrastructure independent.

- How machine learning technologies impact web application security?

To promote discussions, we organized breakout sessions with time to discuss different topics. For each discussion topic, we randomly divided the interested participants into 2 groups to discuss the different challenges. At the end of the discussion, we gathered the two groups and confrontate the summaries of the two groups' breakouts.

In the following, we provide the schedule of the meeting, the list of all talk's abstracts, and the topics and challenges discussed during the breakout sessions.

Overview of Talks

Title: Security and Privacy, Platform vs Engine
John Wilander

Abstract: Browser vendors have to protect both the integrity of the web engine itself and provide a platform which allows web developers to secure their websites and provide user privacy. This talk explores the difference between them and some current challenges. All three full-fledged browser engines use a multi-process architecture. The engine security goal is to contain any exploits of memory corruption bugs and not let them attack the full browser or other websites. This has led to inter-process communication, or IPC, being the primary security boundary to defend since IPC is a deliberate opening in the sandbox. Objects from untrusted code are serialized into byte streams and sent over IPC. The challenge is to safely recreate valid objects on the recipient side under the assumption of a malicious sender.

Site isolation lies in-between engine and platform defense. Whereas it's mostly been talked about as a protection against speculative execution attacks, it also protects against memory corruption bug exploits achieving full read/write/execute privileges.

Anti device fingerprinting also lies in-between engine and platform defense. Neither the engine nor web APIs should reveal fingerprintable information unless absolutely necessary. A challenge here is the difference between entropy and uniqueness. Less entropy, such as the language setting "en" rather than "en-US", is often more unique and thus more susceptible to fingerprinting.

Web platform defense comprises things like CSP, SOP, partitioning of all observable state, and navigational tracking protection. Two challenges here are adoption and incentives.

Title: Evaluating Web Archives for Reproducible Web Security Measurements
Ben Stock

Abstract: Given the dynamic nature of the Web, security measurements on it suffer from reproducibility issues. In this paper we take a systematic look into the potential of using web archives for web security measurements. We first evaluate an extensive set of web archives as potential sources of archival data, showing the superiority of the Internet Archive with respect to its competitors. We then assess the appropriateness of the Internet Archive for historical web security measurements, detecting subtleties and possible pitfalls in its adoption. Finally, we investigate the feasibility of using the Internet Archive to simulate live security measurements, using recent archival data in place of live data. Our analysis shows that archive-based security measurements are a promising alternative to traditional live security measurements, yet reproducible by design. As an important contribution, we identify insights and best practices for future archive-based security measurements.

Title: Privacy breaches by chatbots in group messaging
Hsu-Chun Hsiao

Abstract: New privacy concerns arise with chatbots on group messaging platforms on the web, as chatbots may access information beyond their intended functionalities, such as messages meant for other group members or sender identities. Chatbot operators may

exploit such information to link users across groups and infer personal attributes from messages, potentially leading to web tracking and targeted advertising. State-of-the- art secure group messaging protocols guarantee end-to-end security against platform providers. However, their security guarantees break with the introduction of chatbots. This talk presents the preliminary case studies on current group messaging platforms and highlights the ideas and challenges to develop a privacy-preserving group messaging protocol against adversarial chatbots.

Title: When protecting users breaks the web
Steven Englehardt

Privacy-focused browsers and browsers with a smaller marketshare constantly face website breakage. In this talk, I present the challenges we've faced at DuckDuckGo in detecting and mitigating website breakage via user reports. I show that there's a disconnect between user-perceived breakage and what we think of as a broken site. I also discuss the tension between helping users recover from breakage themselves versus discovering and fixing broken websites for all users. I share how we've had success in changing behavior and improving reports, and point to opportunities for future work.

Title: Ongoing Work: A Systematic Overview of the Challenges in Blackbox Dynamic Application Security Testing
Merve Sahin

Abstract: This is an ongoing work, funded by the EU commission under the project TESTABLE. This work aims to provide a systematic analysis of the challenges related to blackbox web application testing, by making a comprehensive literature review of the relevant publications since 2010. Until now, we identified 57 challenges that are (partially or fully) addressed in the academic literature, and categorized them into four groups (exploration, vulnerability identification, exploitation, and validation.) We also identified the techniques employed in each paper to address the relevant challenge (e.g., symbolic string analysis to overcome the challenge of input filtering), and the types of vulnerabilities that are difficult to discover due to the relevant challenge (e.g., difficulty to discover client-side XSS vulnerability due to sanitizer function). Finally, we aim to provide a benchmark to evaluate the existing blackbox vulnerability scanners, with respect to their ability to overcome the testing challenges identified in the first step. For this, we plan to extend the OWASP Benchmark project with additional test cases. We believe that our systematic approach will guide future research on which problems to prioritize. Moreover, the benchmark will help the security testers to be aware of the potential pitfalls of the dynamic security testing tools.

Title: Understanding and measuring "bad" ads
Lujo Bauer

Abstract: Do online ads and tracking cause harm? Reporting on our own and other researchers' work (much by Eric Zeng), this talk discusses what makes an online ad "bad", and describes several efforts to measure whether some groups of people are likely to be targeted with bad ads. Example results include that even visits to web pages with sensitive content are routinely tracked and that race, ethnicity, and age can correlate with the

likelihood of seeing bad ads. Finally, the talk touches on the challenges of measuring ads at scale.

Title: Security Challenges in Web Systems from the Perspective of Runtime Platform Evolution
Zhenkai Liang

Title: Developer-Centric Approach towards Web Security
Sebastian Roth

Abstract: Security mechanisms or improvements to those have been developed with focus on whether they technically solve the underlying issue. However, examples like PGP encryption for emails have shown that this often does not work in practice. We should in addition to technical solutions start to understand the human as a core part of each technical system. Be it a developer or end user, we need to find roadblocks for security and work together with the affected stakeholders on solving those issues. Besides presenting work that tackles human factors in Web security, this talk will contribute to the discussion about the usability of web security mechanisms and ways how to improve upon the current situation.

Title: Abstract Interpretation-based Static Analysis for Security: Abstractions for Security
Xavier Rival

Abstract: In this talk, we will discuss general issues related to the abstraction of security properties. The design of a static analysis generally requires to fix a semantics, to select an abstraction, defined by a family of logical predicates together with their machine representation, and finally to set up algorithms for abstract operators. In the case of security properties, the second step is made more difficult by the fact that the target properties usually require reasoning over several executions. Therefore, we will present a few existing approaches, discuss their limitations and possible paths to novel abstract domains for security.

Title: Code-reuse attacks in JavaScript-driven server-side applications and runtimes
Musard Balliu

Abstract: The last decade has seen a proliferation of code-reuse attacks in the context of web applications. These attacks target vulnerabilities in which attacker-controlled data exploits legitimate code fragments within a web application’s codebase to execute a code chain that performs malicious computations, for example Remote Code Execution, on the attacker’s behalf. In this presentation, we discuss how large-scale static and dynamic code analysis helps discovering vulnerabilities in high-profile applications and JavaScript runtimes.

Title: Model Driven Security and Privacy
David Basin

Abstract: We review recent research of ours in model driven security and privacy. In our work, one generates web applications, with complete, configured support for enforcing security and privacy policies, from high level models. We report, in particular, on a recent usability study that supports the thesis that developing secure systems from models is feasible and advantageous in practice.

Title: Should it Stay or Should it Go? Generalising and relating Information Erasure and Data Minimisation
David Sands

Abstract: Information erasure is the principle that information should be deleted once it has served its purpose. Data minimization is the principle that you should not collect more information than needed for a specific purpose. In this talk we describe the challenges to unify these concepts in a language-based setting.

Title: Least privilege access for persistent storage in browsers
Abhishek Bichhawat

Abstract: Web applications often include third-party content and scripts to personalize a user's online experience. These scripts have unrestricted access to a user's private data stored in the browser's persistent storage like cookies and localstorage associated with the host page. If some of these scripts behave maliciously, they can easily access and modify private user information like session-id, user consent, etc. that are stored in these storages. The goal of our work is to design an approach to restrict their access to it. Our approach enforces least privilege access for third-party scripts on these objects to ensure their security by attaching labels with the storage objects that specify which domains are allowed to read from and write to these objects on the page. We implement our approach on the Nightly Firefox build and show that it effectively blocks scripts from other domains from accessing the storage objects, as per the policy.

Title: Repairing DoS Vulnerability of Real-World Regexes
Tachio Terauchi

Abstract: There has been much work on synthesizing and repairing regular expressions (regexes for short) from examples. These programming-by-example (PBE) methods help the users write regexes by letting them reflect their intention by examples. However, the existing methods may generate regexes whose matching may take super-linear time and are vulnerable to regex denial of service (ReDoS) attacks. This work presents the first PBE repair method that is guaranteed to generate only invulnerable regexes. Importantly, our method can handle real-world regexes containing lookarounds and backreferences. Due to the extensions, the existing formal definitions of ReDoS vulnerabilities that only consider pure regexes are insufficient. Therefore, we first give a novel formal semantics and complexity of backtracking matching algorithms for real-world regexes, and with them, give the first formal definition of ReDoS vulnerability for real-world regexes. Next, we present a

novel condition called real-world strong 1-unambiguity that is sufficient for guaranteeing the invulnerability of real-world regexes, and formalize the corresponding PBE repair problem. Finally, we present an algorithm that solves the repair problem. The algorithm builds on and extends the previous PBE methods to handle the real-world extensions and with constraints to enforce the real-world strong 1-unambiguity condition.

Title: Jack-in-the-box: An Empirical Study of JavaScript Bundling on the Web and its Security Implications
Cristian-Alexandru Staicu

Abstract: In recent years, we have seen an increased interest in studying the software supply chain of user-facing applications to uncover problematic third-party dependencies. Prior work shows that web applications often rely on outdated or vulnerable third-party code. Nonetheless, existing measurement studies neglect an important software engineering practice: developers often merge together third-party code into a single file called bundle, which they then deliver from their own servers, making it appear as first-party code. Ignoring bundling may result in underestimating the complexity of modern software supply chains. In this talk, we aim to address these methodological shortcomings of prior work. To this end, we propose a novel methodology for automatically detecting bundles, and partially reverse engineering them. Using this methodology, we conduct the first large-scale empirical study of bundled code on the web and examine its security implications. We provide evidence about the high prevalence of bundles, which are contained in 40% of all websites, and the average website includes more than one bundle. Following our methodology, we reidentify 1051 vulnerabilities originating from 33 vulnerable npm packages, included in bundled code. Among the vulnerabilities, we find 17 critical and 59 high severity ones, which might enable malicious actors to execute attacks such as arbitrary code execution. Analyzing the low-rated libraries included in bundles, we discover 10 security holding packages, which suggest that supply-chain attacks affecting bundles are not only possible, but they are already happening.

Cookie Crumbles: Breaking and Fixing Web Session Integrity
Pedro Adão

Abstract: Cookies have a long history of vulnerabilities targeting their confidentiality and integrity. To address these issues, new mechanisms have been proposed and implemented in browsers and server-side applications. Notably, improvements to the Secure attribute and cookie prefixes aim to strengthen cookie integrity against network and same-site attackers, whereas SameSite cookies have been touted as the solution to CSRF. On the server, token-based protections are considered an effective defense for CSRF in the synchronizer token pattern variant. In this paper, we question the effectiveness of these protections and study the real-world security implications of cookie integrity issues, showing how security mechanisms previously considered robust can be bypassed, exposing Web applications to

session integrity attacks such as session fixation and cross-origin request forgery (CORF). These flaws are not only implementation-specific bugs but are also caused by compositionality issues of security mechanisms or vulnerabilities in the standard. Our research contributed to 12 CVEs, 27 vulnerability disclosures, and updates to the cookie standard. It comprises (i) a thorough cross-browser evaluation of cookie integrity issues, that results in new attacks originating from implementation or specification inconsistencies, and (ii) a security analysis of the top 13 Web frameworks, exposing session integrity vulnerabilities in 9 of them. We discuss our responsible disclosure and propose practical mitigations.

Title: Microarchitectural side-channel mitigations for serverless applications
Aastha Mehta

Abstract: Serverless platforms execute application functions from multiple tenants on shared servers. Although the functions are logically isolated in containers or VMs, an adversarial tenant could observe the usage of the shared physical resources of the server (e.g., caches) by a colocated "victim" tenant and exploit the observations to infer the victim's secrets. Prior work has demonstrated constant-time execution as a principled approach to mitigating such microarchitectural side-channel exploits. However, prior techniques have mostly focused on hardening cryptography applications, which are typically written in static languages that compile to native executable code. In contrast, serverless application developers implement functions in dynamic, feature-rich languages, such as javascript and python, where prior constant-time tools and techniques are not directly applicable. We propose to develop an automatic microarchitectural side-channel mitigation tool for javascript applications, which implements constant-time transformation in V8, a popular javascript engine.

Title: Static analysis for web applications: testability challenges and improvements
Luca Compagna

Abstract: This research, funded by the EU commission under the project TESTABLE, focuses on static analysis for web applications. We introduced the concept of testability patterns for SAST (static application security testing), aka code obstacles that, when present, impede the ability of state-of-the-art static analyzers to properly scan the web application code. We showed how SAST tools struggle with these patterns (less than 50% support), how these patterns are prevalent in the real world (in average one obstacle every 20 LoC), and how these patterns can be remediated via code refactoring, improvements for the SAST tools, and novel SAST strategies. Among these novel strategies, we introduced the WHIP, the first approach that enables SAST tools to “collaborate” by sharing information that can help them to overcome each other’s limitations. We launched an OWASP project to share our research with the web security community and to create a community around this topic: https://owasp.org/www-project-testability-patterns-for-web-applications/. Overall, our results indicate that even companies using commercial SAST tools in their software development life cycle may get a false sense of security, as many code areas may just be uncovered by the used SAST tools.

Title: Hardening the Firefox Web Browser
Christoph Kerschbaumer

Abstract: Today, the vast majority of applications are vulnerable to code injection attacks (XSS). In this talk we will explore techniques that have proven successful to eliminate certain types of code injection attacks within the Firefox codebase. Ultimately, there is no silver bullet that can eliminate all vulnerabilities in applications, but we will examine how using a layered defense strategy allows us to harden Firefox against injection attacks.

Title: Security implications of the File System Access API
Dolière Francis Somé

Abstract: Modern browsers' security is the foundation for a safe online experience. While traditional browser compromises (e.g., sandbox escapes due to memory-related bugs) are becoming relatively rare, browsers are simultaneously adding increasingly more powerful APIs for interacting with the local machine, which can be misused to escape the browser's sandbox. In this work, we show how oversights in the specification and implementation of one such API—the File System Access API—can be used to deploy powerful man-in-the-browser attacks or exfiltrate user data. Our attacks target browser profiles—the persistence layer of browsers, which stores user preferences and data on the disk between sessions. We show that, except for Tor, all modern browsers use this feature, but they do little to protect the integrity and confidentiality of stored profiles. Hence, we discuss how web attackers can alter browser profiles via the File System Access API to install malicious browser extensions in Google Chrome. We also present cross-browser attacks against the Firefox browser to install a malicious root certificate authority, redirecting all HTTPS traffic to a man-in-the-middle proxy server and silently enabling the camera, microphone, or GPS to spy on the user. We argue that the newly proposed File System Access API fundamentally changes the web threat model, enabling untrusted JavaScript code to behave like traditional malware executing directly on the user's machine. We responsibly disclosed the issues to the vendors, who acknowledged and fixed many of them. We conclude by discussing how to harden browser profiles against the demonstrated attacks.

Title: Towards ethical server-side web scanning
Ben Stock

Abstract: Comprehensive and representative measurements are crucial to understand security and privacy risks on the Web. However, researchers have long been reluctant to investigate server-side vulnerabilities at scale, as this could harm servers, disrupt service, and cause financial damage. This can lead to operator backlash and problems in peer review, as the boundaries posed by the law, ethics, and operators' stance towards security research are largely unclear. In this paper, we address this research gap and investigate the boundaries of server-side scanning (3S) on the Web. To that end, we devise five typical scenarios for 3S on the Web to obtain concrete practical guidance. We analyze qualitative data from 23 interviews with legal experts, members of Research Ethics Committees, and website and server operators to learn what types of 3S are considered acceptable and which behavior would cross a red line. To verify our findings, we further conduct an online survey with 119 operators. Our analysis of these different perspectives shows that the absence of

judicial decisions and clear ethical guidelines poses challenges in overcoming the risks associated with 3S, despite operators' general positive stance towards such research. As a first step to mitigate these challenges, we suggest best practices for future 3S research and a pre-registration process to provide a reliable and transparent environment for 3S-based research that reduces uncertainty for researchers and operators alike.

Meeting Schedule

	Sun, Mar 17	Mon, Mar 18	Tue, Mar 19	Wed, Mar 20	Thu, Mar 21
7:30 AM		Breakfast	Breakfast	Breakfast	Breakfast
9:00 AM		Welcome and Introduction			
9:30 AM					
10:00 AM		Talks	Talks	Talks	Discussions
10:30 AM		Break	Break	Break	Break
11:00 AM					
11:30 AM		Talks	Tutorial	Tutorial	Talks
12:00 PM		Lunch	Lunch	Lunch	Lunch, end of seminar
1:30 PM		Photo	Social events, no meeting		
2:00 PM					
2:30 PM					
3:00 PM		Discussions		Discussions	
3:30 PM	check-in?	Break		Break	
4:00 PM					
4:30 PM					
5:00 PM					
5:30 PM		Discussions		Discussions	
6:00 PM		Dinner	Banquet	Dinner	
7:00 PM	Banquet				
7:30 PM					
8:00 PM					
8:30 PM					

Discussions and summary of future directions

Topic: Browser infrastructure for academics

Lead: Abhishek Bichhawat

The discussion topics included the following questions:

- What do we (how do we) use a browser for in academic research?
- Which browser do we use the most for research or analysis?
- Are the issues that we research browser specific?
- Would a minimalistic browser be helpful for advancing research in the area with a proof-of-concept or are real-world browsers the way to go about it?
- Is instrumenting a browser part of the research or are extensions enough for attaining the same goals?
- What support/collaborations from the industry/organizations can help academics working with the browsers?
- What academic research is useful for a browser vendor to invest resources in?

As part of the discussion, the following references arose as a way to find how different browser implement a specification:

- https://caniuse.com/
- https://mozilla.github.io/standards-positions/
- https://privacytests.org/
- https://wpt.fyi/results/

Topic: Web measurement infrastructure sharing and maintenance and result replicability

Lead: Ben Stock

The discussion topics included the following questions:

- What are the key issues towards replicating results?
  - Geolocation / server IPs
  - Login State? Browsing history/state
  - Depth of crawl
  - Interaction?
  - Time of crawl?
  - Type of devices (headless, headfull, etc)
- Should we tolerate a certain level of error? How can we quantify these in the first place?
- To what extent should we even aim for replicability of results? The Web is extremely ephemeral anyways.
- Should we rely on external services to provide replicability (e.g., Archive)? Seems unfit for many purposes

- How can we build privacy-preserving (e.g., blinded review) data sets which can be re-used?
- What should our stance be on making infrastructure available?
- Maintenance seems to be a gigantic pain, the only real research tool (which is reused) seems to be OpenWPM?
- Does using OpenWPM for replicability kill "real-world" views (Market share of Chrome is much bigger)?

Open Research Questions

- How do you define "errors" in measurements? How do you define equality?
- What is the minimal format for storing data in such a way that others can still work with it (not just dumb WARC files)?
- What is the impact of crawl parameters (browsers, depth, visit lengths, scrolling) on different types of measurement results?
- Should we have high-fidelity data to sanity-check the code available for replication?
- What can we learn from adjacent fields (e.g., network analysis) or even unrelated fields (astronomy)?

Topic: Usability of web security mechanisms

Lead: Lujo Bauer

Starting points for the discussion:

- Web standards conversations – well established that web is for end users first
    - shouldn't get tied up in building only for devs, or business, or browser devs
- GPC: opt-in flag – is that what people are asking for?
- end-users can be completely lay-people or privacy aware
- tools
    - mozilla observatory – will scan domain and report on how well it uses security features
        - example of a usable tool, but at a moment in time
        - https://observatory.mozilla.org/, but relaunching via MDN https://developer.mozilla.org/en-US/blog/mdn-observatory/
    - Google CSP evaluator
        - https://csp-evaluator.withgoogle.com/
    - SSL Server test
        - https://www.ssllabs.com/ssltest/

- problems/challenges/wishes
    - features and standards don't come with development tools – community is missing default baseline implementations
    - dev tools vendors should steer devs away from deprecated APIs, known bad code
    - frameworks should come with safe default
    - firefox/mozilla had a "recommended" tag for extensions

        - https://support.mozilla.org/en-US/kb/recommended-extensions-program
        - but developers don't want to be judged
            - can we have positive feedback only?
    - most users don't have the ability to assess whether an, e.g., extension is good or bad from a security perspective (even at the spec level?)
        - users might want to have a single rating of privacy risk
        - https://privacytests.org/
            - but to be universally accepted the source needs to seen as impartial
        - https://www.eff.org/pages/secure-messaging-scorecard
    - What are the incentives to make their tools usable?
    - user-facing interfaces (e.g., about numbers of trackers) seem to be going away
        - maybe because users didn't have a way of improving the situation
    - any interest in more interactive tools? (copilot style?)
    - Are there psychology studies about how many options should be given to the user?
        - acceptable level of required interaction depends on user knowledge, interest, and context (are they doing a security/privacy thing or not)
    - How can we incentivise the usage of security mechanisms?
- Do we need a Web Security Indicator for Websites? (similar to the lock icon)
- Should we hide new APIs behind security requirements? (similar to API levels in the mobile domain)
- Should we make Security as an opt-out (e.g. CORS)?

- research ideas
    - "sanitizer" that compiles browser in such a way that memory allocated to different websites is protect
- takeaways
    - web security researchers should more often work with usable security & privacy researchers
    - there is space for independent raters of browser security features (privacytests.org, but not working for brave and weighting checkmarks), and for developing weightings for passing/failing tests
        - https://ssd.eff.org/
    - web standards community should provide web IDE developers lists of APIs to deprecate / recommend what should be used
    - dev tools should provide safe defaults

Topic: New grand challenges in web security

Lead: John Wilander

Objectives

- Share research in the area (papers, tools, implementations)
- Identify new solution ideas

Navigation without URLs
Problem:
Navigation from source.example to destination.example with a URL like this: https://destination.example/path/page.html?userID=639668262. The userID is data transfer and not needed to load the page/resource,

Goal:
Be able to navigate cross-site without personalized data transfer between the two.

Two threat models:

- No collusion between source.example and destination.example
    - … other than *inclusion* of cross-site script under destination.example
- Collusion between source.example and destination.example

Side effect-free rendering

Problem:
Side effects of webpage use can be tracked to profile the user. Side effects can be network requests or state change like setting JavaScript variables.

Goal:
As much webpage browsing as possible without side effects.

Two threat models:

- Remote attacker — side effects seen by a server
- Local attacker — side effects seen in local state

Privilege model for JavaScript

Problem:
All scripts in the first party context have equal privileges. Thus they can steal/leak sensitive data, rewrite page content etc.

Goal:
Lower the privileges of non-first party scripts.

Three threat models:

- Injected scripts
- Deliberately included scripts that are compromised
- Deliberately included scripts that are not compromised but intently …
    - Violate user privacy
    - Attack competitors
    - Use resources, e.g. mine cryptocurrency

Topic: New and emerging threat models of web applications

Lead: Christoph Kerschbaumer

Most critical security risks to web applications (OWASP Top 5):

1. **Broken Access Control:** Access control enforces policy such that users cannot act outside of their intended permissions. Failures typically lead to unauthorized information disclosure, modification, or destruction of all data or performing a business function outside the user's limits

2. **Cryptographic Failures:** The first thing is to determine the protection needs of data in transit and at rest. For example, passwords, credit card numbers, health records, personal information, and business secrets require extra protection

3. **Injection:** An application is vulnerable to attack when user-supplied data is not validated, filtered, or sanitized by the application.

4. **Insecure Design:** Insecure design is a broad category representing different weaknesses, expressed as “missing or ineffective control design.

5. **Security Misconfiguration:** The application might be vulnerable if the application is missing appropriate security hardening across any part of the application stack or improperly configured permissions on cloud services.

Links and Information:

- https://owasp.org/Top10/
- Hardening Firefox against Injection Attacks
- Enforcing Content Security by Default within Web Browsers

Research Opportunities:

- Code Injection Detection and Data Exfiltration Mechanisms
- Security By Default Design of Web Applications
- Encrypt Everything (Data in Transit, Local Storage, Cloud Storage)
- Build better tooling (similar to mozilla observatory) which tells Web applications what to look out for.

Topic: Formal methods and web security

Lead: David Sands

Discussed questions:

- What are exemplary cases of formal methods in web security?
- What problems/areas should we target? What are the bottlenecks?

- What areas are dead-ends for FM? (cost-benefit tradeoff, over-complex real systems, ...)
- What new methods or models need to be developed?

References discussed:

- From Research Prototypes to Continuous Integration: Guiding the Design and Implementation of JavaScript
    - https://blog.sigplan.org/2023/01/12/from-research-prototypes-to-continuous-integration-guiding-the-design-and-implementation-of-javascript/
    - https://github.com/es-meta
    - ECMA-SL https://github.com/formalsec/ECMA-SL2
- WebAssembly
    - Wasm SpecTec: Engineering a Formal Language Standard
        - https://arxiv.org/abs/2311.07223
    - Concolic Execution for WebAssembly
        - https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ECOOP.2022.11
        - https://github.com/formalsec/wasp
- Web Infrastructure Model
- Formalizations and proofs of TLS, e.g. F*, Tamarin
- Verified Crypto: HACL* integrated to Mozilla

Topic: JavaScript sandboxing

Lead: Cristian-Alexandru Staicu

Context for the discussion:

- Scenario 1: Topic 3 in the Grand challenges session - JavaScript code from different parties should run in different contexts, with different capabilities.
- Scenario 2: there are several non-browser JavaScript runtimes like Node.js, Deno, or Bun (https://twitter.com/wesbos/status/1756029879487791239). Most of them support some form of lightweight sandboxing by providing an API for isolated execution of code, as pioneered by Node.js' vm module. However, as a result of prior work, this API is marked as "not a security mechanism.

To be useful, both these types of sandboxes must *offer tight integration with the host code, e.g., by allowing pointer sharing and full mediation* of powerful builtin APIs, i.e., intrinsics. There is a TC39 proposal in Stage 1 for providing such a solution to the JavaScript world: https://github.com/tc39/proposal-ses, built on top of this Stage 2 proposal: https://github.com/tc39/proposal-shadowrealm.

Discussion points:

- Are there concrete use cases in which this form of sandboxing is justified? Known use cases: TAP platforms, smart contracts, protect against misbehaving libraries (supply chain attacks); Figma, Google, Yahoo and Facebook before they all decided it is too dangerous and they should use iframes instead.

- Is this the revival of web mashup security: [1], [2], [3], [4], [4]? If so, why did we stop working on these in the first place?
- Are we just missing some stronger security mechanisms in the non-browser environments (iframes, CSP)?
- Should more powerful isolation techniques (workers, processes) also allow tighter interactions between the host and the guest code? (call by reference vs. call by values, shared pointers, side effects).
- Should a policy language be standardized for whitelisting APIs/resources in JS/web sandboxes and runtimes?
- There are some recurring pitfalls/misuses when interacting with sandboxes that allow whitelisting or reference sharing. Is there a generic way to avoid them? How can we assist developers in writing better policies?
- Beyond JavaScript sandboxes: are the APIs for isolating and interacting with native extensions and web assembly appropriate? Do we need a one-size-fits-all solution?
- Should JS/web sandboxes consider side channels? Should they provide a way to access lower-level primitives (OS kernel modules, TEEs)?
- Should sandboxes perfectly mimic the host environment, e.g., to have same intrinsics? If not, some JS code might become environment-specific, e.g., by detecting it is running inside a sandbox.
- What is the performance impact of sandboxing? Cost in battery might be important
- Site isolation is slow, but everybody needed to implement it because the threat was too serious. Security vs. performance.
- Still lacking a power use case and/or a serious threat. Possible use cases:
    - Privacy is the main drive for implementing such a feature
    - Server-side/Electron-like environments JS might be a power use case
- Alternative proposal: always know the origin of a particular piece of JavaScript inside the engine
- Tracking origin can be quite slow, however, you only need one bit of taint for every instruction
- Usability issues might be a problem
- Idea: throttle ads/untrusted scripts
- Assume the people will do the right thing in security, instead of privacy.
- Bringing Web API to Wasm and sandboxes using permissions might not be a priority. Web Assembly is mostly used for performance
- Workers introduced their own problems (botnets) even though the sandboxing was quite sturdy.
- Lockdown mode might be a good example of how such a feature might be deployed
- Benchmarks for performance regression might not be good enough
- This might be a very niche feature for the web that we still need
- Vm2 escapes , is SES a secure isolation mechanism?

Topic: ML and web security research

Lead: John Mitchell

This session will invite discussion on the security of current and future AI-enabled web applications, focusing on the parallel between traditional code injection attacks and new prompt injection attacks that subvert the behavior of language models used in web applications. To provide a basis for this discussion, we will begin with a short review of recent progress in AI and some of the broader research directions related to AI security. We will then consider web application security by looking at the architecture of current applications that may pre-process user input, pass the result to an AI model, and then post-process the output. This is analogous to the way that conventional web applications

may pre-process user input before querying a database and then post-process the result. With this application architecture in mind, participants will discuss sample prompt injection attacks and possible defenses. Time permitting, we will also discuss emerging theories of AI-model behavior and ways that such theories could allow us to reason usefully about the security of AI-enabled web applications.

Topic: Web security education

Lead: Musard Balliu

Context: Education is one of the most important contributions that we researchers make to society. Cybersecurity education in general and web security education in particular are becoming increasingly important in light of existing and emerging threats. These developments pose the challenge of providing meaningful education that strikes a good balance between foundational aspects and practice.

Discussion points:

- What are the core and emerging topics that we should teach in a web application security course?
- Web application security is typically a single module in a security course - time to think about full courses in web application security?
- Attacks are cool, defenses are boring: how to increase interest in building secure software?
- What infrastructures do we use to teach web security? How to consolidate the effort?
- How does web security education change in the new era of AI/ML?