Poster: HybridGuard: A Principal-based Permission and Fine-Grained Policy Enforcement
Framework for Web-based Mobile Applications

(Recently published research)

P. H. Phung, A. Mohanty, R. Rachapalli, and M. Sridhar. HybridGuard: A Principal-based Permission and Fine-grained Policy Enforcement Framework for Web-based Mobile Applications. In Proceedings of the IEEE Workshop on Mobile Security Technologies (MOST), May 2017.

ABSTRACT:

Web-based or hybrid mobile applications (apps) are widely used and supported by various modern hybrid app development frameworks. In this architecture, any JavaScript code, local or remote, can access available APIs, including JavaScript bridges provided by the hybrid framework, to access device resources. This JavaScript inclusion capability is dangerous, since there is no mechanism to determine the origin of the code to control access, and any JavaScript code running in the mobile app can access the device resources through the exposed APIs. Previous solutions are either limited to a specific platform (e.g., Android) or a specific hybrid framework (e.g., Cordova) or only protect the device resources and disregard the sensitive elements in the web environment. Moreover, most of the solutions require the modification of the base platform.

In this paper, we present HybridGuard, a novel policy enforcement framework that can enforce principal-based, stateful policies, on multiple origins without modifying the hybrid frameworks or mobile platforms. In HybridGuard, hybrid app developers can specify principal-based permissions, and define fine-grained, stateful, and history-based policies that can mitigate a significant class of attacks caused by potentially malicious JavaScript code included from third-party domains, including ads running inside the app. HybridGuard also provides a mechanism and policy patterns for app developers to specify fine-grained policies for multiple principals. HybridGuard is implemented in JavaScript; therefore, it can be easily adapted for other hybrid frameworks or mobile platforms without modification of these frameworks or platforms. We present attack scenarios and report experimental results to demonstrate how HybridGuard can thwart attacks against hybrid mobile apps.

Link to published paper:
https://www.computer.org/csdl/proceedings/spw/2017/1968/00/1968a147-abs.html

HybridGuard: A Principal-based Permission and Fine-Grained Policy Enforcement Framework for Web-based Mobile Applications

Phu H. Phung[1], Abhinav Mohanty[2], Rahul Rachapalli[2] and Meera Sridhar[2]

1. University of Dayton, Ohio 2. University of North Carolina at Charlotte

HYBRIDS APPS ARE HERE TO STAY!

Ionic Survey [>13k developers, 2017]

- Only 2.9% developers exclusively using native only tools
- 32.7% to completely abandon native development in favor of hybrid

StackOverflow Survey [>64k developers, 2017]

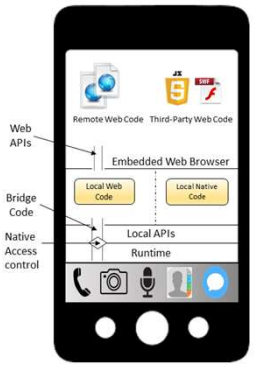
- 73% reported as web developers
- ~32% targeting hybrid/web-based and Progressive Web Apps (PWA)

Some Popular Hybrid Apps

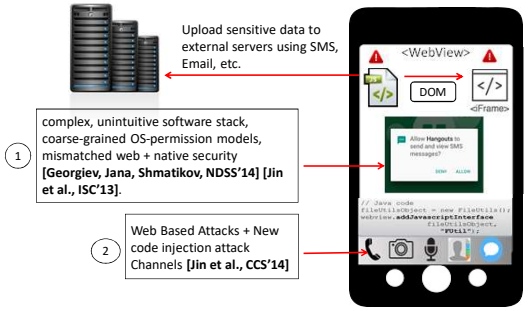
- FINANCIAL TIMES
- GMAIL (iOS)
- LINKEDIN
- GOOGLE DOCS (iOS)
- UBER
- EVERNOTE

Cordova, jQuery mobile, Onsen UI, PhoneGap, Intel XDK, Xamarin, React Native, Sencha, ionic

HYBRID APP ARCHITECTURE



HYBRID APP ATTACK SURFACE & EXISTING DEFENSES



- OS-based access control
  - Limitation: coarse-grained; once permission is granted (static or dynamic), no control over how app uses permissions
- Domain Whitelisting, e.g., in Cordova
  - Limitation: ad networks have to be whitelisted; their loader scripts can be tampered with and made malicious
- Content Security Policy
  - Limitation: removing `eval()` can break functionality of many apps
- Same Origin Policy
  - Limitation: JS bridges added to browser by local code, have no web origin as far as the browser is concerned. Therefore, malicious web content from any origin can directly invoke the bridges.

HYBRIDGUARD: MAIN CONTRIBUTIONS

In-lined reference monitoring (IRM) [**Fred B. Schneider, TISSEC'2000**] framework for hybrid mobile apps to mitigate against real-world attacks.
Threat Model : Third Party JavaScript (JS)

- mediation of security-relevant bridge and DOM API
- principal-based access control
- fine-grained, history-based security policies

HYBRIDGUARD: ENFORCEABLE POLICY CLASSES

Resource bounds Policies

Access to `Geolocation` [NYTimes ads], e.g., "Limit to 1 per hour"

Whitelist Policies

Access to `Geolocation`, e.g., "Limit sending the location to a list of origins or IPs"

History-based Policies

Access to `Network`, e.g., "No Network Send (Internet, SMS, etc.) after `Storage` Access"

Custom Policies

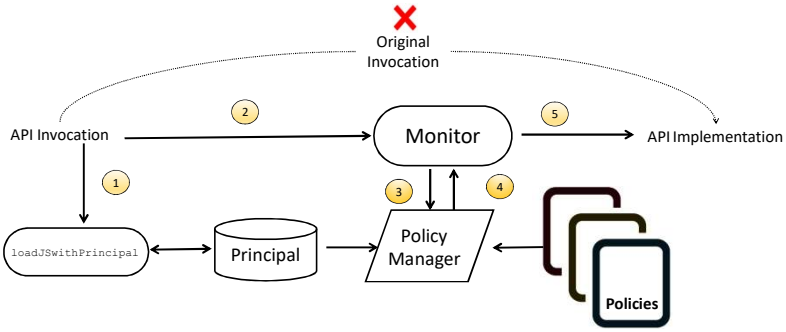
e.g., "If an app requires SMS registration, the app should send just a single SMS."

EXPERIMENTAL RESULTS ON REAL-WORLD HYBRID APPS

Application Name	Resource Accessed	Policies Enforced	Example Policy
`Parked Car Locator`	`GeoLocation`	WP	WP: The policy implemented is giving Geolocation access to "maps.google.com" but not "ad.leadbolt.net"
`Fan React`	`Contacts SMS`	HBP,WP,RBP	HBP: SMS should not be sent immediately after reading contacts
`Graded`	`Contacts File System SMS`	HBP,WP,RBP	RBP: Contacts must be read only once per day
`Remote SMS Control`	`Contacts File System SMS`	HBP,WP,RBP	WP: SMS should be sent to the number that controls devices
`Web Ratio`	`Contacts File System`	HBP,WP,RBP	HBP: 1. After reading QR code, there should not be file or contact read. 2. Config file should be read only once a day.
`My Car Navigator`	`Geo Location Accelerometer`	WP,RBP	WP: Geo Location should be accessed by application website only.

Resource bounds Policy (RBP) Whitelist Policy (WP) History-based Policy (HBP)

OVERVIEW: HYBRIDGUARD APPROACH



1. JavaScript (.JS) files are loaded by our API and assigned a principal for runtime monitoring
2. Invocation to the wrapped APIs are forwarded to the monitor
3. Monitor consults the Policy Manager to verify permissions and enforce security policies
4. Policies are pre-defined by the developer in a JSON file and accessed by the Policy Manager
5. No Violation = Access to original API invoked

FUTURE WORK

- IRMs for mitigating native side attacks in Hybrid Apps
- Static Analysis for inferring of an equivalence class of permission region in single page apps
- Formal Policy Spec language for Hybrid App IRMs
- IRMs for Embedded browser in native apps

MAIN REFERENCES

- M. Georgiev, S. Jana, and V. Shmatikov. Breaking and Fixing Origin-Based Access Control in Hybrid Web/Mobile Application Frameworks. In Proceedings of the 21st Annual *Network and Distributed System Security Symposium* (*NDSS*), 2014.
- X. Jin, X. Hu, K. Ying, W. Du, H. Yin, and G. N. Peri. Code Injection Attacks on HTML5-based Mobile Apps: Characterization, Detection and Mitigation. In Proceedings of the 2014 *ACM SIGSAC Conference on Computer and Communications Security* (*CCS*), pages 66–77, 2014.
- X. Jin, L. Wang, T. Luo, and W. Du. Fine-grained Access Control for HTML5-based Mobile Applications in Android. In Proceedings of the 2015 *Information Security Conference* (*ISC*), pages 309–318. Springer, 2015.
- P. H. Phung, A. Mohanty, R. Rachapalli, and M. Sridhar. HybridGuard: A Principal-based Permission and Fine-grained Policy Enforcement Framework for Web-based Mobile Applications. In Proceedings of the *IEEE Workshop on Mobile Security Technologies (MOST)*, 2017.
- F. B. Schneider. Enforceable Security Policies. In Proceedings of the *ACM Transactions on Information and System Security*, 3:30–50, *TISSEC'2000*.
- M. Shehab and A. AlJarrah. Reducing Attack Surface on Cordova-based Hybrid Mobile Apps. In Proceedings of the 2nd *International Workshop on Mobile Development Lifecycle* (*MobileDeli*), pages 1–8, 2014.

PLEASE CONTACT : MEERA SRIDHAR (msridhar@uncc.edu)

For complete work, see publication in: MOBILE SECURITY TECHNOLOGIES (MOST) 2017
2017 IEEE Security and Privacy Workshops (SPW)