Cached and Confused: Web Cache Deception in the Wild

Seyed Ali Mirheidari¹, Sajjad Arshad², Kaan Onarlioglu³, Bruno Crispo¹, Engin Kirda², William Robertson²

¹ University of Trento, ² Northeastern University, ³ Akamai Technologies
Web Caches

scheme://user:password@host:port/path?query#fragment
Path Confusion

scheme: \texttt{user:password@host:port/path?query#fragment}

→ URL rewriting mechanisms: Clean URLs (a.k.a. RESTful URLs)
  ◆ Web servers interpret URLs in ways that are \textit{not clearly reflected} in the externally-visible part of the URL string.

\textbf{Web Server:} \url{http://example.com/index.php/v1} => \url{http://example.com/files/index.php?p1=v1}

→ Browsers, caches or proxies are \textit{not aware} of this abstraction

\textbf{Other Components:} \url{http://example.com/index.php/v1}

→ What about: \url{http://example.com/index.php\n%2Fv1%2Ffake.css%3F%23fake.css}?
  ◆ Browsers & CDNs can get more \textit{confused} with customized encoding URL!
Web Cache Deception (WCD)

WCD: Different interpretations of a URL (path confusion) between a server and a cache.

1. GET /account.php/nonexistent.jpg

2. 200 OK
   Cache-Control: no-store
   <account.php> (!)

3. GET /account.php/nonexistent.jpg
   200 OK
   <account.php> (!)
Research Questions

➔ How common is WCD on popular, high-traffic sites?
➔ What is the impact?
➔ Can variation of Path Confusion expand the number of vulnerable sites?
➔ Are CDN’s vulnerable by default?
Appended “/<random>.css” to each URL from the victim account.

Visited same page from the (un)authenticated attack crawler and compare responses.

Novel Path Confusion techniques applied to the attack URLs
Results

➔ 50K vulnerable pages in 37 sites out of 349 (10.7%)!
   ◆ Personally Identifiable Information (PII), Security tokens, session identifiers and authorization keys leaked on vulnerable pages.

➔ Sophisticated attack scenario using WCD.
   ◆ CSRF token bypass, session hijacking, XSSI, OAuth Covert Redirect, etc.

➔ Proposed novel Path Confusions are quite effective to confuse most of CDNs.
   ◆ Increased detection rate by 45%.

➔ Voted and led to an award as Top Web Hacking Technique of 2019 by Portswigger!

Lessons Learned: System Safety Problem

➔ WCD is a “system safety” problem.
   ✦ There are no isolated faulty components.
   ✦ There is no complete solution such as a Hotfix.

➔ Mitigation remains a cross-functional responsibility.
   ✦ Complex interactions among different technologies should be evaluated.
   ✦ Examining not only individual system components but also their interactions.
   ✦ Reviewing how vendor configurations interact with internal systems!
Conclusions

➔ WCD: The origin server and cache disagree about cacheability.
➔ WCD can impose critical risk to the system!
➔ We developed a repeatable methodology to discover WCD.
  ◆ ~11% of tested sites were vulnerable!
➔ WCD impacts all cache technologies.
➔ Caching rules based on file extensions are prone to security problem.
➔ Path confusion techniques make it possible to exploit 45% more sites.
➔ There is a widespread lack of user awareness.
➔ CDNs are not intended to be plug & play solutions.
Thanks! Questions?

Seyed Ali Mirheidari, seyedali.mirheidari@unitn.it
Sajjad “JJ” Arshad, @sajjadium