Security Advisory 2019-001

Web Cache Poisoning Vulnerabilities

January 23, 2019 — v1.0

TLP:WHITE

History:

- *23/01/2019 — v1.0 – Initial publication*

Summary

Web cache poisoning has long been considered a *theoretical* threat. However, already published research describes practical examples of this type of attack [1]. Also, recently there have been documented cases of observing exploitation of these types of vulnerabilities on production systems.

Technical Details

Caching improves web-page load times by reducing latency while also reducing the load on application server. It can be implemented at different levels: specific software, offered by content delivery networks (CDN), or built-in into web applications and frameworks. All of these are susceptible to cache poisoning.

Web cache poisoning is a specific type of a more generic family of cache poisoning vulnerabilities [2]. The impact of a maliciously constructed response from a webserver can be magnified if it is cached and served to multiple users. The published research [1] presents practical ways of cache poisoning by using **unkeyed inputs**. Unkeyed inputs are parts of a request that a cache does not use for *mapping* the caches.

Products Affected

Cache servers and services, web applications and frameworks.

Recommendations

- Disable caching, if possible from operational point of view. In some cases caching is enabled by default – not necessarily needed for performance reasons.
- If disabling the cache is not possible, restrict caching to purely static responses.
- Audit every URL of an application with `Param Miner` to detect and disable unkeyed inputs. `Param Miner` is a `Burp Suite` extension used to detect unkeyed inputs [3].

References

[1] https://portswigger.net/blog/practical-web-cache-poisoning

[2] https://www.owasp.org/index.php/Cache_Poisoning

[3] https://github.com/PortSwigger/param-miner