source

DEFENSE

# APPROACHES TO MITIGATING WEBSITE SUPPLY CHAIN ATTACKS

source
DEFENSE

# 1. Introduction and Executive Summary

Supply chain attacks are an attractive and popular technique for malicious actors to compromise an organization's technological infrastructure. By targeting discrete components of a complex system, attackers can exploit the relatively less-sophisticated security countermeasures in place around a vendor-supplied component of the overall system.

Although supply chain attacks are well known in many cybersecurity contexts (e.g., payment skimming, industrial infrastructure, etc.), this technique is less readily apparent within the context of customer-facing web properties. Due to the nature of contemporary customer-facing web properties, a variety of vendor-supplied components are commonly used in concert to deliver the desired user-behavior, interactivity and ultimately monetization to satisfy business requirements and drive customer satisfaction.

# 2. Attack Vector Overview

Web sites rely heavily on 3rd party vendors to provide functionality and behavior expected of visitors. 3rd parties provide analytics, user behavior analysis, customer engagement, display advertising and, in general, monetization of a website's content and brand. Although 3rd parties provide immense value, they also introduce a broad security risk to the visitor of an organization's web site.

JavaScript, the programming language which enables interactivity on the web and facilitates the functionality of these 3rd party tools has very poor inherent security controls. In fact, it would be fair to say that there are no controls available at all. Every line of JavaScript which is executed by a web browser has the same level of access, permission to run and interaction with the user that any other line does. The consequence of this is that there is no way to control the behavior of any particular script within any given web session using standards-based approaches.

Each script in a traditional computer programming context may be thought of as its own application running within the browser. Unlike traditional applications, however, each piece of JavaScript is accessed in real-time as the visitor requests a web page. There is no concept of compilation, bundling or package delivery within the browser -- each script arrives fresh on each pageview with whatever behavior it contains at that particular moment.

The combination of these factors leads to a serious security flaw, namely, that 3rd parties which drive monetization can and do introduce unintended behavior into the visitor experience. And, if a third party is compromised, that unintended behavior may take the form of malicious exploitation: credit-card skimming, defacement, PII disclosure, etc.

In essence, an enterprise hosting content through a web server is extending unlimited trust to each 3rd party as it relates to the visitor's experience. In the context of modern cybersecurity, this is untenable from the perspective of due diligence, reputation management and compliance.

source
DEFENSE

# 3. Possible Approaches

## Usage restriction

### Overview

Restricting the use of third-party resources is a rational and effective way to address the risk introduced by a third-party into a the end-user's browser. By reducing the number of possible points of ingress into the browser session one can effectively narrow opportunities for compromise. Combining this technique with thorough review of third-party code the regular monitoring of those parties' behaviors within the browser session can alleviate much of the concern they pose to the end-user's experience and security.

### Implementation

Usage restriction is an organizational or procedural technique which is not defined by a specific technology or technologies. It may consist of simply reducing the number of third party scripts in use on a page without review or consideration, thus mitigating risk but not preventing it, or it may combine reduction in count with review and control, which provides further mitigation but introduces overhead. A third-party usage restriction methodology may be constructed with the input from several parts of an organization. Some constituents to consider may be:

- Security
- Marketing
- Content owners
- Content stakeholders
- Systems administrators
- Developers

Invariably each perspective within an organization will have different requirements with regard to how narrow a scope can be defined in terms of third-party inclusion.

From a pure security standpoint, the elimination of all third parties is obviously ideal, however, a content stakeholder may desire to include as many as necessary and to incorporate them as quickly as possible to drive monetization. Developers in turn may need certain third-party functionality to facilitate good development practices, however, they may view responsibility and accountability for those third parties to be operationally cumbersome.

**Benefits**

Usage restriction can provide a curative effect for the issue of third-party risk when considered from a security perspective. Without introducing the vector into the end-user's environment, the problem can be effectively eliminated.

**Disadvantages**

Along within eliminating risk, usage restriction may also cut-off avenues to monetization or efficiency. An organization driven by audience conversion or brand development will essentially require the use of third parties within their web properties to function in an efficient and financially viable manner, for instance. In another example, service-based organizations may find customer satisfaction reduced by excluding functionality expected by consumers in a modern context, such as customer support chat, social media integration or rich media.

Also important to consider is operational burden associated with usage restriction. A mixed-use or partially restrictive policy would require due diligence on behalf of the organization with regards to those resources allowed under such a policy.

**Conclusion**

Usage restriction is a valid and valuable approach to this problem. It is, however, operationally expensive and in many cases simply untenable from a business perspective. Therefore it is likely that most organizations will find usage restriction on its own as an unrealistic way to interact with third parties.

## Content Security Policy

**Overview**

Content Security Policy (CSP) is a technology which provides additional security for resources loaded into a web page at the time it is rendered within a browser. CSP can define what type of resources may be loaded from particular domains and provides a mechanism for the end-user's browser to report violations of set policies. In essence, CSP allows the administrator of a website to define with whom that website communications. CSP has existed as a draft specification for sometime, however, adoption by most common web browsers was not widespread until around 2015.

## Implementation

CSP, when considered from the standpoint of implementation, is a server-side technology. To implement CSP a server administrator would configure a web server to respond to each request from an end-user's browser with the standard content related to an HTTP request and additionally a set of headers related to the CSP policies for that content.

Aside from simply enabling CSP within the origin webserver, a server administrator or person responsible for the content delivered from that server must define the policies themselves which are to be served along with the content.

In brief, a policy as defined in the context of CSP is comprised of:

- the type of content to be loaded by the page
- the domain or domains from which that content may be loaded and optionally,
- a reporting mechanism for violations.

These policies may be applied to any response the web server returns or may be narrowed to a subset of pages, subpaths, etc. CSP can also control outbound connections from the browser and some more additional attributes of content.

## Benefits

CSP provides security beyond that which was possible within the browser prior to its introduction. From the context of web security, it offers additional resiliency against attempts at cross-site scripting or other, cross-origin-based attacks.

WIth regards to 3rd party risk, CSP can help by providing a list of known-trusted sources with each HTTP response or request and the server administrator can positively affirm that only those sources will appear within the end-user's browser. Additionally, the reporting mechanism defined by the CSP standard offers further insight into the behavior of content in the end-user environment than what has traditionally been available.

## Benefits

There are several considerations a server administrator, application developer or content owner would need to consider during the implementation of CSP.

First, the underlying nature of CSP's policy enforcement may or may not be suitable depending on the application of the technology. CSP essentially operates on the same model as a classic whitelist / blacklist approach towards access control and suffers from similar shortcomings:

- Tusted sources must be known at the time of configuration
- Administrators responsible for the configuration of policies must have an intermediate-to-advanced knowledge of those content sources to ensure behavior functions as intended
- Communication must be maintained owner or maintainer of the content source for version control and possible changes in domains
- Any partnerships between the site's third-parties and their partners (fourth-parties) will be blocked by such a policy as this content tends to be dynamic and change from user to user, making it impossible to predict.
- Policies cannot discern between malicious and benign content from a trusted source

The first two points are similar in nature in that they both may contribute to the operational overhead of implementing CSP in even a moderately-sized environment.

Additionally, these two factors may prove challenging for organizations which separate the concerns of server administration from content administration. As server technologies, particularly web content and application servers, become more specialized and atomized within an organization's technical infrastructure there is a widening gap between those responsible for ensuring that the content is served and those responsible for knowing what the content actually is. CSP requires both parties to work in concert to ensure that effective policies are created, enforced and monitored for effective application.

Finally, CSP's whitelisting approach to access management is problematic unless one can implicitly trust their known sources of content. A policy as defined through CSP cannot control the behavior of a third-party resource loaded into the browser, only its origin. Because of this, an organization must extend complete trust to any third-party permitted into the browser via CSP. If, for any reason, a third-party that is known and trusted starts delivering malicious content, CSP will not serve as a method of protection or detection.

**Summary**

Content Security Policy is an important and incremental step towards a standards-based approach for mitigating the impact which a compromised third-party may have on the end-user browser environment.

For specific applications, such as ensuring that content is appropriately loaded from trusted sources within an organization but different than the origin (i.e., two subdomains within the same organization), CSP may be sufficient to prevent many attacks. For environments with diverse, dynamic sources of third-party content, however, CSP's operational overhead and reliance upon trust make it a less desirable or potentially insufficient altogether.

## Subresource Integrity

**Overview**

Subresource Integrity (SRI) is a technological approach to handling third party resources within a browser environment. SRI allows for browsers to check loaded resources against a known-valid hash tag associated with that resource to ensure that it has not been modified or tampered with between the time the hash was generated and the resource was transmitted to the browser.

**Implementation**

SRI is implemented on the document-level. A document served to a browser must include both references to the resources to be loaded and a hash associated with those resources.

At some point in time, the author of the document or the web application serving the document would need to generate hashes, or strong cryptographic digests, for each resource to be referenced and loaded within the end-user's browser. When the browser attempts to load the requested resources, it will compare the hash of the resource against the resource itself and determine whether there is match, indicating that the resource has not been modified since the time the has was generated.

Resources which do not match their associated hash will not be loaded by the browser. SRI may be combined with CSP to provide more granular assignment of SRI as a requirement.

## Benefits

SRI, or really any hashing technique, ensures that a content author can provide a reliable mechanism for content consumer to verify they have received the content as intended. Cryptographic hashing, document signing and other related technologies are a cornerstone of many security technologies. They are robust and reliable techniques to ensure the integrity of information and facilitate a trustless environment between the end-user and the content author.

## Disadvantages

SRI presents some disadvantages when considered in the context of mitigating malicious third-party content. The primary disadvantage of SRI is that it is a point-in-time assessment of the resource; it relies on that resource not changing between the time of authorship and the time of use. For many applications of this technique, such as static content like cryptographically signed messages, this is an appropriate and even desirable behavior.

For third-party JavaScript resources, however, this technique will prove almost impossible to use. A prerequisite condition to any hash method, is that the file being hashed remain static; most if not all of the third-party services offered today provides a tailored response for a specific user; meaning thier JavaScript will change from user to user. Moreover, web technologies change rapidly. Innovations in continuous integration and product delivery mean that the consumers of those technologies may be using revisions and subversions of JavaScript applications may happen many times per week, if not more frequently.

Applying an SRI-based security model to this sort of environment requires constant updating of cryptographic hashes and a very close relationship between the organization and the third party.

## Conclusion

SRI is a useful technology for validating integrity of the data requested from and communicated to the browser, however, side effects of a hash-and-check approach make it less useful when relying on third-parties who may publish frequent revisions to the resources they offer.

source
DEFENSE

# Summary and Conclusions

3rd party risk presents, in many ways, a novel challenge to traditional enterprise security strategy. Because of the combination of clear business necessity, poor security architecture and rapidly accelerating exploitation, the attack vector presented by 3rd party JavaScript within the browser requires a unique approach and careful consideration from any organization providing content to visitors.

Standards-based approaches towards mitigating this attack vector are well-engineered and logically sound, however, they fail in the sense that they take the perspective of a web application developer or maintainer. In other words, technologies like CSP and SRI work well in the context of a self-developed application: if you know everything about how your application works then surely you can know what other code it incorporates and how that code functions. You may even be deploy technologies like dynamic application testing or application monitoring to further secure that application. This, however, is not the challenge presented by 3rd party JavaScript.

The landscape of a contemporary customer-facing website is wholly unlike an internally developed web application. The participants contributing code in an average visitor's browsing session number in the dozens, if not more. As such, it is impossible for an enterprise to know, let alone control, the entirety of the attack surface.

Other techniques such as application monitoring, usage restriction, code review and general due diligence are valid approaches, however, they rely heavily on operational and business expense. In other words, it is possible for organizations to implement these techniques but that effort will expend resources and decrease revenue. Each of these approaches fundamentally rely on three things: time, effort and talent. Should an organization decide that they possess the sufficient surplus of resources in those three categories then these may be desirable avenues to pursue. Given the finite and often restrictive constraints of security resources, however, most enterprise organizations may find themselves unprepared to respond to this emerging threat.

In summary, it may be possible to partially address the risk presented by 3rd party vendors through traditional approaches, but only at great cost to the organization and with limited effectiveness in terms of mitigation. Unfortunately, traditional security technologies and techniques are proving to be an insufficient response to this emergent and accelerating threat.