May 29, 2019

Serverless Cybersecurity Training

Serverless Cybersecurity Training

Historically, training students and professionals in cybersecurity practices is expensive, as it requires setting up and maintaining complex laboratory environments where the trainees learn how to attack and defend targets (e.g., vulnerable computers) in a contained environment. The contained environment may consist of tens or even hundreds of servers running software with various security vulnerabilities. The trainees attempt to find and exploit the vulnerabilities while possibly also fixing or patching the vulnerabilities to prevent others from exploiting them. As emphasis on cybersecurity has increased, these training programs have become increasingly popular.[1] This paper offers a solution to reducing the cost of running these cybersecurity lab environments by eliminating the actual lab altogether. Instead, a simulated lab environment is created within the confines of each trainee's browser by leveraging the latest technologies that are available in modern browsers.

Current technologies that run safe and controlled laboratory environments can be divided into two groups: (1) on-premise solutions; and (2) cloud-based solutions. On-premise solutions execute locally on a machine on the trainee's network.[2] These solutions are downloaded and then executed by a local server. Cloud-based solutions, on the other hand, execute on cloud instances.[3] Cloud-based solutions may start a separate vulnerable virtual machine for each trainee. Each of these solutions have significant drawbacks. For example, they may require

[1] See https://www.cybrary.it/become-penetration-tester/ and https://bootcamp.berkeley.edu/cybersecurity/landing/
[2] See https://github.com/WebGoat/WebGoat and https://www.vulnhub.com/
[3] See https://google-gruyere.appspot.com/ and https://www.paloaltonetworks.com/services/education/cybersecurity-skills-practice-lab

significant bandwidth (e.g., downloading an entire virtual machine), technical expertise, or the use of specialized software.

This paper discloses using Hypertext Markup Language 5 (HTML5) Service Workers and Web Workers to run simulated servers within the confines of the trainee's browser without actually attacking or defending a server. First, the trainee or user may be instructed to visit a specific domain hosting the exercises. In this example, the domain is exercise-domain.com. This site exclusively serves static files (i.e., files that do not need to be generated, modified, or processed before transmission), and thus can be deployed cheaply, as static files are easily cacheable and consume minimal computing resources. As illustrated in Figure 1 below, the domain (i.e., exercise-domain.com in this example) starts an HTML5 Service Worker (*S*) in the user's browser. A Service Worker essentially acts as a proxy server that sits between the browser and the network. The Service Worker *S* intercepts all subsequent HTTPS network requests (*R*) directed to exercise-domain.com from the browser. The Service Worker *S* then redirects the intercepted network requests *R* that were targeting exercise-domain.com instead to an HTML5 Web Worker (*W*).

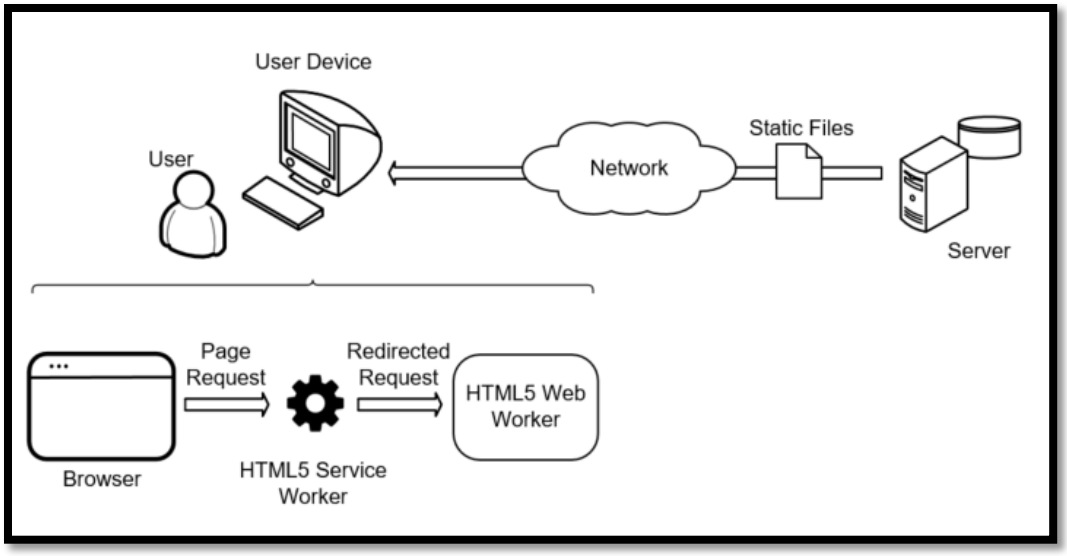

Figure 1

The Web Worker *W* contains the business logic of the exercises. For example, the Web Worker *W* may contain a vulnerable image, document, database, or any other service. The Web Worker *W* executes JavaScript (or some other similar high-level language). For example, the Web Worker *W* may run an operating system compiled in JavaScript or a webserver that executes JavaScript.[4] The Web Worker *W* replies to the user's network request *R* by sending a response to the Service Worker *S*. The Service Worker *S* then provides the response back to the browser (and therefore the user). In this manner, the browser believes that it is communicating with the domain exercise-domain.com, while in reality, the network request *R* never left the user's device. This can be extended to any number of domains to recreate a cybersecurity laboratory experience, where multiple domains are running vulnerable servers that can be attacked or patched to prevent attacks.

[4] See https://github.com/copy/v86

The use of Service Workers and Web Workers in the manner provides a number of distinct advantages. The method decreases the cost of running a cybersecurity laboratory considerably as the use of only static files reduces the cost to that of running a personal website. Furthermore, the method isolates the work of each trainee. Because trainees can often exploit services in unexpected ways, and in fact, this is frequently encouraged by the training, the services may crash or fail, affecting the experience of other trainees. Traditionally this is solved by generating a separate laboratory environment for each trainee which rapidly increases cost. The method also limits liability. That is, offering vulnerable machines exposed on the internet can be risky, as trainees could exploit and repurpose the machines to, for example, attack other sites.

The described method only exists in the user's browser and therefore does not provide any valuable advantage to a trainee for attacking other online services. The method also allows trainees to inspect and interact with the inner workings of running vulnerable services to exploit or patch them. Because each trainee remains fully isolated in a secure environment, patching a vulnerable service will not affect other trainees and the validity of the patch may be tested by a script running in the user's browser. The user machine may be any machine that can execute a modern browser. Therefore, trainees are not limited to just traditional desktop computers and may use, for example, mobile devices. Additionally, training can be performed offline, as the Service Worker *W* may cache the static files from the server locally.

The methods described may be used in anywhere a virtualized lab environment is useful. For example, computer science training in various topics (cybersecurity, distributed computing, full stack development) or product demonstrations (e.g., where a sales representative wants to create an ephemeral lab environment to demonstrate a service).

ABSTRACT

Training students and professionals in cybersecurity practices is expensive due to the complex laboratory environments required to properly isolate the trainees. Traditional methods include downloading and executing a virtual machine locally or accessing cloud-based virtual machines. In this work, this cost is reduced by obtaining static files from server that cause a user's browser to execute an HTML5 Service Worker and a Web Worker. The Service Worker intercepts browser requests to one or more domains and redirects them to the Web Worker. The Web Worker, which provides the vulnerable services the trainee interacts with, responds to the user.