­­Vineet: Internship Projects

**Project 1: API Security at Scale**

Background: APIs are first class citizens in Adobe’s microservices-first architectures. Their “micro”-ness has caused explosion in their numbers and hence AppSec team must find ways to secure APIs at scale. We not only need visibility into API behavior but also scalable and concrete ways to find undesired behavior. The intern project will touch on some key aspects of the vision for API security at scale.

One concrete line of exploration is generation of a Security spec to capture how the service is doing authorization and authentication, what are the inputs (eg query params, post body, etc) that a service is receiving, what are the data types for these inputs, as so on. In essence it can be thought of as an OpenAPI spec with some additional fields having some security context.

The current status is that we are working on generating the spec from a collection of request-response (e.g., HTTP archive - HAR). The idea is that a collection of requests – responses will capture enough information about APIs (e.g., a typical system test session would consist of functional (happy path, exceptional), performance, stress tests, etc.).

This spec will serve as a snapshot of API behavior. It would have data such as authentication or authorization mechanisms which will help determine if these mechanisms are correctly implemented, well-formed requests for fuzz testing, attribution information and point of contacts in case we find issues. These are just some of the use cases for the spec. As we have create a collection of specs, there will be many more use cases for the spec.

Project ideas:

* SpecGeneration Module: Takes a collection of HTTP requests-responses and generates a security spec.
  + Intern Project: Improve existing implementation
    - Understand, re-write (as needed)
    - New feature: Capturing oAuth spec
      * First understand oAuth. One good reference is “oAuth 2 in action” book.). A service can be authZ server, resource server or client in the oAuth (understand differences and security weaknesses in each). Propose what we need to capture in the spec.
      * Implement this module
    - Add unit tests with different HAR files
* SpecAnalysis Modules: Takes a spec and performs security analysis.
  + Intern Project: Develop different analysis modules:
    - Syntax checking aka Spec well formedness – Verify if a spec has all required data or if it is missing some sections
    - Semantics based analysis –
      * Missing AuthN from requests – e.g., no tokens in requests with side effects (POST, PUT, DELETE, etc)
      * OAuth traffic issues – e.g., missing CSRF/state token in client to AuthZ server traffic.
    - Think of more analysis we can do here and add them
    - Add unit tests for the analyses
* Stretch Goal: Package the modules as a standalone tool which teams can use, deploy the tool on Adobe infrastructure, expose the functionalities via endpoints. This would allow working with systems used by product teams at Adobe such as IMS, Adobe IO, etc

Resources & Background Preparation

* <https://swagger.io/specification/> - this is the entire specification. A simple overview of what the specification looks like and what it contains is good enough. In depth understanding is not necessary
* <https://www.medianova.com/en-blog/2019/07/15/a-step-by-step-guide-to-generating-a-har-file> - how to generate a HAR file
* <https://wiki.python.org/moin/PyUnit> or any other Python unit test framework
* <https://apisecurity.io/encyclopedia/content/api-security-encyclopedia.htm>
* OAuth 2 in Action: https://learning.oreilly.com/library/view/oauth-2-in/9781617293276/

**Project 2: Contribute to SecuritySDK modules**

Backgroud: SecuritySDK is a collection of modules which are used by many different automation projects. Some examples – a JIRA module to automate creation of tickets in JIRA using the API, a properties manager module to manage properties like secrets (getting them from Vault, getting them from config files, etc)

Project idea: Add new modules (currently there is no fixed plan on what can be added)

Resources: Good knowledge of Python should be good enough.

**NOTE:** The project ideas are not fixed, and some nuances might change – but the overall big picture will likely remain the same.