To: Professor Krasso

From: David Tarvin

Date: 21 Jun 2019

Subject: Security Recommendations for Microservices

Now that we have decided to implement and deploy microservices in this company, we need to address the security of such services. I have outlined three recommendations for security which I believe to be a jumping off point. There are many more issues and strategies we can look at for securing our microservices, but I believe the three recommendations I’ve made here are the ones we should implement first.

My first recommendation is that we adopt OAuth 2.0 as our standard for user authentication. OAuth2 is currently the industry standard. OAuth 2.0 gives us four choices of how a user can get an access token: through authentication with an “authorization server”, through a redirect from a third-party site to a URL containing an access token, or through using the username and password from a third party site (such as Google) to obtain a token. This gives us much flexibility in setting up our user authorizations. Admittedly, OAuth is usually vulnerable to four forms of attack: lack of data confidentiality and server trust, insecure storage of secrets, OAuth implementation with flawed session management, and session fixation attack with OAuth. However, several companies have built solutions for improving the security level of OAuth, plus there are several libraries and platforms for OAuth that we can use to speed up our development phase.

My second recommendation is that we adopt a defense in depth protocol for our more sensitive services. Defense in depth means that we apply several layers of security to those services, so an attacker would have to overcome not just one but all of our defenses to access those services. This can include building fine-grained firewalls between each service, with sets of rules and host-to-host encryption. The fact that we are using microservices allows us to set more fine-grained security, such as service-specific rules to containers, APIs and firewalls.

My third recommendation is that we analyze our data stores component-by-component and decide which ones (if any) need encrypting. One nice thing about microservices is that the data for each service is independent of other data, and we don’t have to carry the overhead of encrypting unimportant data just so we can encrypt the important data.

As I said before, we can do so much more besides the tasks listed above in securing our services. However, we need to start somewhere, and these recommendations will be best implemented as we are setting up and deploying the microservices, whereas other strategies can just as easily be added in later. By taking action on security now, we can feel some measure of safety in deploying our services and taking them public.

References

Marco Troisi. (n.d.). 8 Best Practices For Microservice App Sec (article). Retrieved from <https://techbeacon.com/app-dev-testing/8-best-practices-microservices-app-sec>

Johanna Curiel. (n.d.). The State Of Social Authentication: Is OAuth Up To The Job? (article). Retrieved from <https://techbeacon.com/security/state-social-authentication-oauth-job>

Carrie Wodehouse. (n.d.). Tips For Securing A Modernized Service-Based Architecture (article). Retrieved from <https://www.upwork.com/hiring/development/microservices-security/>