DevOps Security Assessment

April 2021

Contents

1 Security assessment

1.1 Risk Identification

Risk identification is the process were we determine risks that could potentially prevent the program from achieving its objectives. Our risk identification includes determining assets and the threats they might face.

1.1.1 Assets

Our assets are as listed:

- Web frontend
- API backend
- Database
- Proxy
- Grafana
- Prometheus
- ElasticSearch
- Filebeat
- Three Droplets

1.1.2 Threat sources (19/4/21)

Our system has at least a couple of vulnerabilities that we are aware of.

1. Our database is not behind a firewall, this means that anyone can try to authenticate with the database.
2. We do not have anything like fail2ban setup on the database server, which means that a brute-force attack could be carried out.
3. We only use basic authentication which provides very little security and are therefore vulnerable to brute force and dictionary attacks.
4. We have three droplets in total. One way of getting access to those droplets is through SSH with a certificate.

1.1.3 Threat sources (5/19/21)

We have added a firewall to help protect our databases. Vulnerability scanning from OWASP ZAP showed us one more vulnerability.

1. Our add_message/ route does not use CSRF tokens in the HTML submission form, which leaves our users vulnerable to CSRF attacks on that route.

1.1.4 Risk scenarios

Based on the threats we might face, we have created risk scenarios as follows

1. Attacker gains access to the elk stack
2. Attacker performs a successful brute-force attack on the web or API service to gain control over one specific user's data.
3. Attacker performs a successful brute-force attack on the database to gain control over all user data.
4. Attacker gains a copy of a private SSH certificate to the droplets, allowing the attacker to SSH into any of them.
5. Attacker gets the victim to click a link that creates a malicious request to add_message/, allowing the attacker to post a message to the victim's account (if the victim is already logged into a session).

1.2 Risk Analysis

Based on our risk scenarios we can now analyse our threats which includes the likelihood of these being realised as well as a look into the impact these threat will course.

Scenario 1

Attacker gains access to the elk stack.

We determine the likelihood of this scenario as almost certain, based on the fact that this has already happened. This has to our knowledge not had any impact till now, but we see the possible impact to be significant. Gaining access to the elk stack, would mean access to our logging. With the given information in this system, it would be possible to understand other parts and therefore increase the risk of further attacks.

Scenario 2

Attacker performs a successful brute-force attack on the web or API service to gain control over one specific user's data.

As stated earlier we do not have any password verification in place, which means that it is possible to choose a one character password or similar. This means that a brute-force attack is almost certain to be successful. This would have a moderate impact on our system by allowing an attacker to authenticate as a specific user of their choice.

Scenario 3

Attacker performs a successful brute-force attack on the database to gain control over all user data.

Brute-forcing on our database is a very unlikely to be successful because we chose a very safe password, making the threat rare. The impact it could do if anyone got access to our database is at least extensive. Without backups, this could simply change or delete all of our data without anyway of getting it back.

Scenario 4

Attacker gains a copy of a private SSH certificate to the droplets, allowing the attacker to SSH into any of them.

The likelihood of anyone obtaining an SSH certificate is very unlikely and therefore we deem it rare. The problem here is the significant impact it could cause if someone got access to all of our droplets. Doing so would compromise our whole system.

Scenario 5

Attacker performs a CSRF attack on a backend user, forcing the end user to execute unwanted actions while authenticated.

The likelihood of a CSRF attack on our backend is unlikely, since the attacker must go through the trouble of using social engineering to get their victim to execute a malicious backend request. This would have a negligible impact on our system by allowing an attacker to post messages to the victim's account.

1.2.1 Risk matrix

As the team moves further with the project, we have to prioritize the risks that we have identified. We use the risk matrix to prioritize the scenarios based on significance.

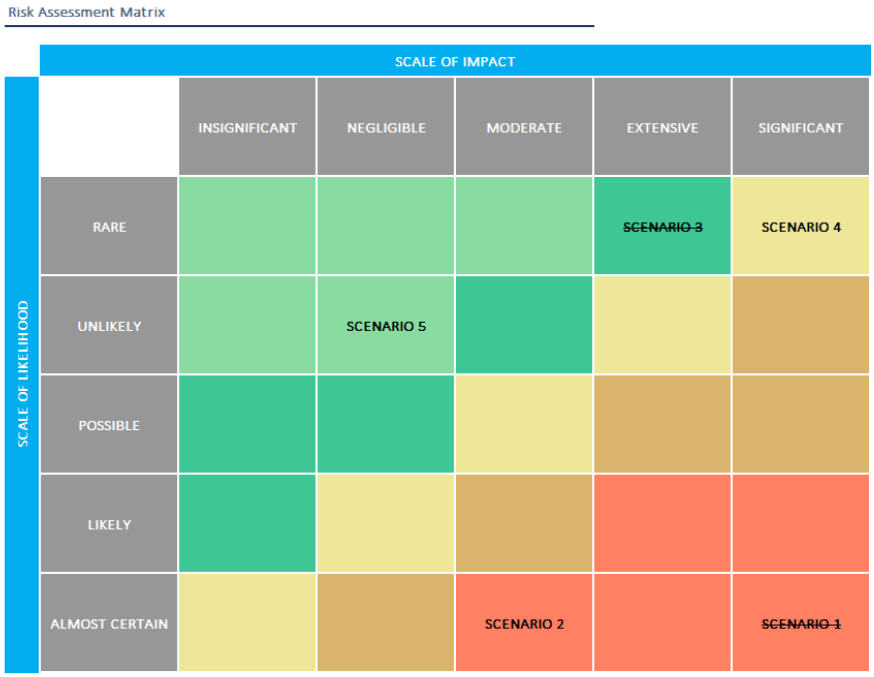

Figure 1: Risk matrix, including each of the scenarios (updated)

Based on the matrix (e.g. figure 1) we can see that it is preferable to prioritize scenario 1, 2 and 3, since they provide the most urgent combinations of impact and likelihood.

1.2.2 Risk Handling

We have already come up with different solutions to each of our scenarios, this includes solutions we prioritize to implement as well as some we do not deem

valuable at this point.

1. For our problem regarding access to the elk stack, we have already seen this happen and are therefore determined to prioritize this. We are sure that setting up a firewall will eliminate the threat that we see as of this moment.

2. Regarding the threat of brute-force attacks on the web or API, we intend to setup password validating to hinder this as much as possible.

3. Regarding the threat of brute-force attacks on the database, this will also require a firewall.

4. One way of hardening the access to the droplets would be to use a password with the SSH certificate. As the threat is rare, we do not see this as a priority as of now.

5. To protect our backend from CSRF attacks, we can use Django's CSRF middleware to protect our API views.