Group C
Esben Gade Hellmuth, Emilie Rafn-Hamalainen, Sean Wachs & Rasmus Dilling Møller
eshe@itu.dk emir@itu.dk sewa@itu.dk rdmo@itu.dk

Monitoring their SLA

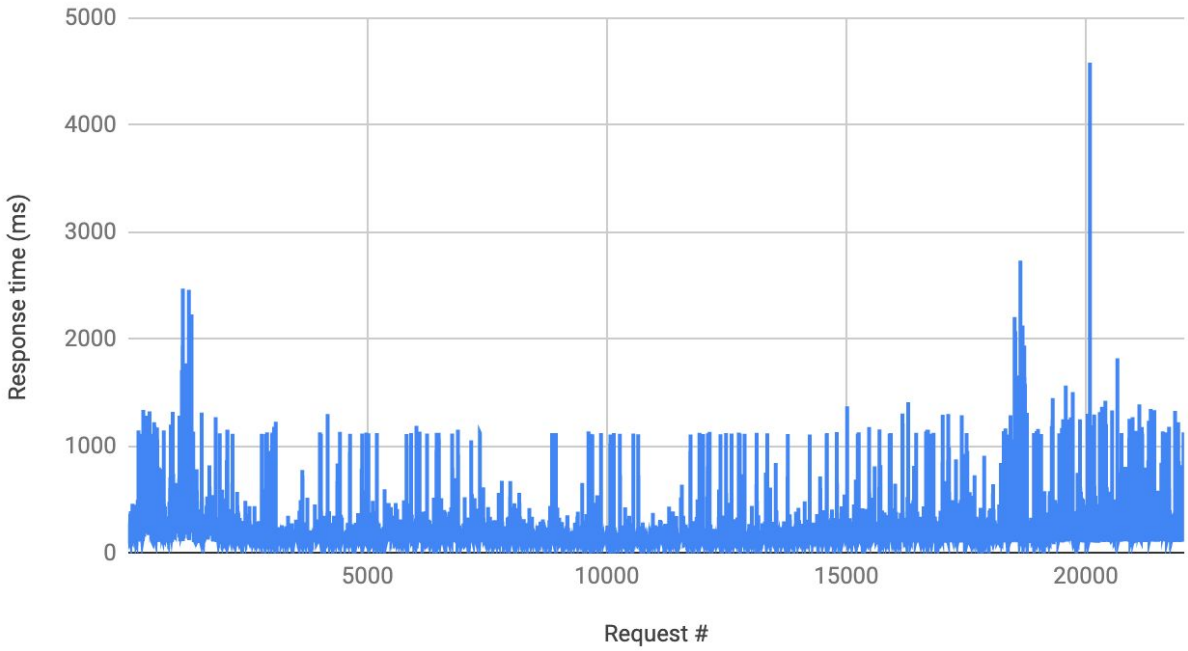
To gather data on their uptime and response time, we ran a script that sent a request every 5 seconds for 24 hours to the api endpoint "https://minitwit.tk/api/latest". After gathering data we did some calculations to see if their service complies with their SLA.

Their SLA says:

- 99.5% uptime average per month.
- 30 minute response time on down (business hours only)
- 500ms max return time

The statistics we calculated:

0 of the requests we sent returned a status code that was not 200 which means that they had a 100% uptime while the script ran.
443 requests out of 22052 (total # requests) had a response time above 500 ms which is 2,01% of all requests. This does not correspond to their SLA. It is not known if they actually meant that it is the average return (response) time that should be max 500 ms.
If this was the case then it would live up to the expectations of the SLA because the total average response time was 172,73 ms.
The graph below gives a visualization of the response time of all requests.



Group C
Esben Gade Hellmuth, Emilie Rafn-Hamalainen, Sean Wachs & Rasmus Dilling Møller
eshe@itu.dk emir@itu.dk sewa@itu.dk rdmo@itu.dk

Testing their security

Port scanning

We started out by running a port scan against their server IP with nmap:

PORT	STATE	SERVICE
22/tcp	open	ssh
80/tcp	open	http
135/tcp	filtered	msrpc
139/tcp	filtered	netbios-ssn
443/tcp	open	https
445/tcp	filtered	microsoft-ds
548/tcp	filtered	afp

As seen in the result their firewall is tight. The only open ports are for ssh and http(s). We attempted to login with ssh, and found that the ssh server only accepts key authentication, not password. The other services in the result are microsoft services which could indicate that they are running a .NET app (okay, we cheated and looked in their code).

A more detailed scan showed that the server's OS is an unspecified flavor of Linux, and that they have Traefik running on port 80. Traefik is an open source edge router/load balancer thingy written in Golang. Cool!

Vulnerability scan

Using Metasploit and WMAP with all modules loaded (except the one that doesn't work), to run a vulnerability scan against the server, gave us zero(!) potentially exploitable vulnerabilities.

Web security

We tried both basic SQL injection and basic Cross-Site Scripting (XSS) but without success.

Group C
Esben Gade Hellmuth, Emilie Rafn-Hamalainen, Sean Wachs & Rasmus Dilling Møller
eshe@itu.dk emir@itu.dk sewa@itu.dk rdmo@itu.dk

Testing our security

Port scanning

PORT	STATE	SERVICE
22/tcp	open	ssh
80/tcp	open	http
3000/tcp	open	ppp

Doing the same port scan on our own server, we realized that we had left port 3000 open by accident. The only intended way to reach Grafana, which is the service we run on port 3000, is through nginx on port 80. We quickly closed the port, but expect to see it in the report on our system's security by our "attacker" group.

A second scan showed us the intended results:

PORT	STATE	SERVICE
22/tcp	open	ssh
80/tcp	open	http

Vulnerability scan

Using Metasploit and WMAP with all modules loaded (except the one that doesn't work), to run a vulnerability scan against our server, gave us some potential vulnerabilities:

```
[*] + [142.93.162.43] (142.93.162.43): directory /Login/
[*]   directory Directory found.
[*]   GET Res code: 200
[*] + [142.93.162.43] (142.93.162.43): directory /login/
[*]   directory Directory found.
[*]   GET Res code: 200
[*] + [142.93.162.43] (142.93.162.43): directory /logout/
[*]   directory Directory found.
[*]   GET Res code: 200
[*] + [142.93.162.43] (142.93.162.43): directory /public/
```

```
[*]   directory Directory found.
[*]   GET Res code: 200
[*] + [142.93.162.43] (142.93.162.43): directory /register/
[*]   directory Directory found.
[*]   GET Res code: 200
[*] + [142.93.162.43] (142.93.162.43): directory /test/
[*]   directory Directory found.
[*]   GET Res code: 200
```

For some reason WMAP reports these as being directories even though they are not. All we got from this is that we should delete an old test site we had forgotten about.

Web security

Our site is protected against SQL injections because we have used prepared statements. We have not done anything to protect against Cross-Site Scripting (XXS).

We have set up our logs so it is easy to see if the log is from the web page or the API, by writing “API endpoint {*then the endpoint*}” before the message, whereas the web page just has a simple error message.

When looking at our logs we can see that there are a lot of logs with the status code 400 around 18:30 Wednesday the 15th of April from the web page. A lot of these logs are from registering, where it has been tried many times in a row to register with either an invalid email, invalid password, or invalid username.

It is not possible to see the tried usernames, passwords, or emails (unfortunately) in the logs so we can not be sure that this has actually been an attempted attack or just a person who is pretty confused about how to register a user.

This has made us aware of making better logs that for example could write the tried usernames and emails. For security reasons, from our users’ perspectives, it would not be a good idea to write the tried passwords in the logs in case of users just forgetting to put in a capital letter, for instance, in their password.

Group C
Esben Gade Hellmuth, Emilie Rafn-Hamalainen, Sean Wachs & Rasmus Dilling Møller
eshe@itu.dk emir@itu.dk sewa@itu.dk rdmo@itu.dk

Risk Assessment Matrix

\ Severity Likelihood \	Negligible	Marginal	Critical
76% - 100%	MEDIUM	HIGH	HIGH
51% - 75%	MEDIUM	MEDIUM	HIGH
26% - 50%	LOW	MEDIUM	MEDIUM
0% - 25%	LOW	LOW	MEDIUM

Asset: User information - **Risks:** Leaking information, Losing information

Given the fairly good results of the security audit of our system, we feel confident that a leak of user information is of low likelihood. However, if it should happen it would be critical. For that reason we have assessed the risk as MEDIUM.

As we deleted our database by accident, we can say with certainty that the likelihood of losing user information is high. When it happened it was critical to our service, and we only recovered because new users signed up and used the service. We have assessed the risk as HIGH.

Asset: System availability - **Risks:** System being unavailable

If our system were to be down at some point, the severity would be Marginal as no information would be lost but it would just not be possible for our users to register/login and post tweets. The likelihood would be low, as our SLA says that our uptime is 99% which means that 1% of the time the system would be down ie. unavailable. Furthermore our SLA also says that the average time to recover is 2 hours, which is the amount of time the system would be unavailable. Therefore the risk is assessed as LOW.