CYBER THREAT DEFENSE

Web Application Penetration Test

Case Study

SpaceCube
ARCHITECTS

***Note: No real company details have been used for this case study**

Introduction

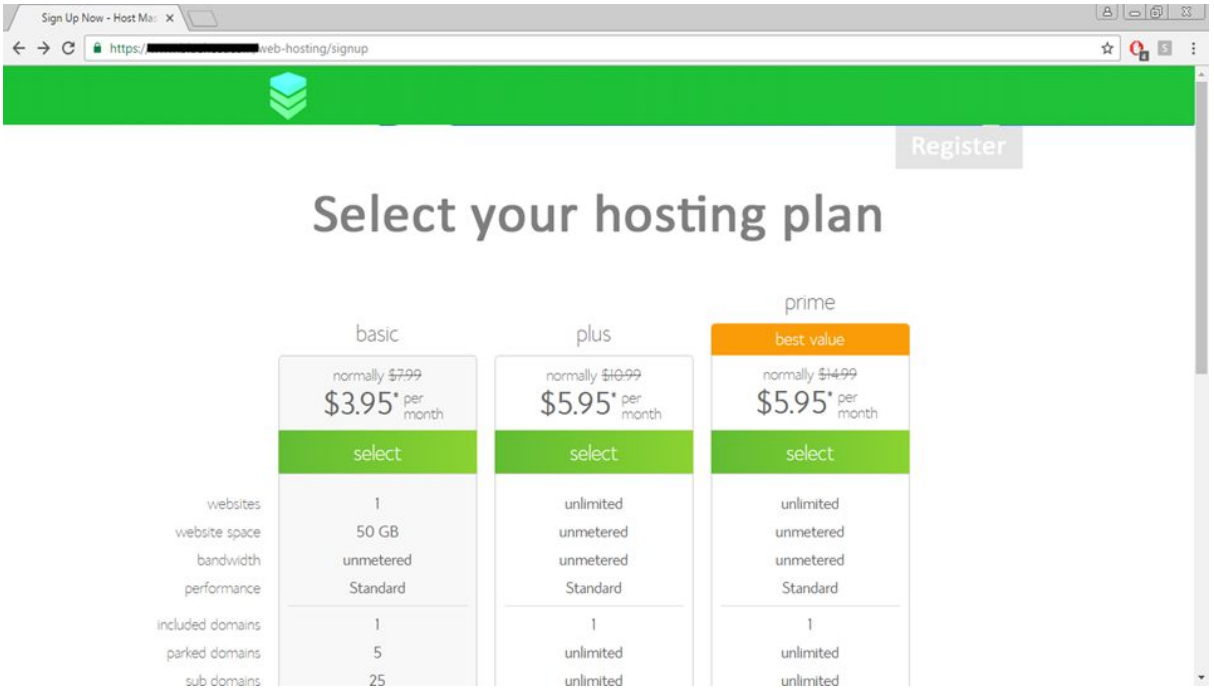
A client that had a SaaS developed to manage server hosting systems, requested an external penetration test against the application, a report with all the vulnerabilities discovered and the remediation solutions.

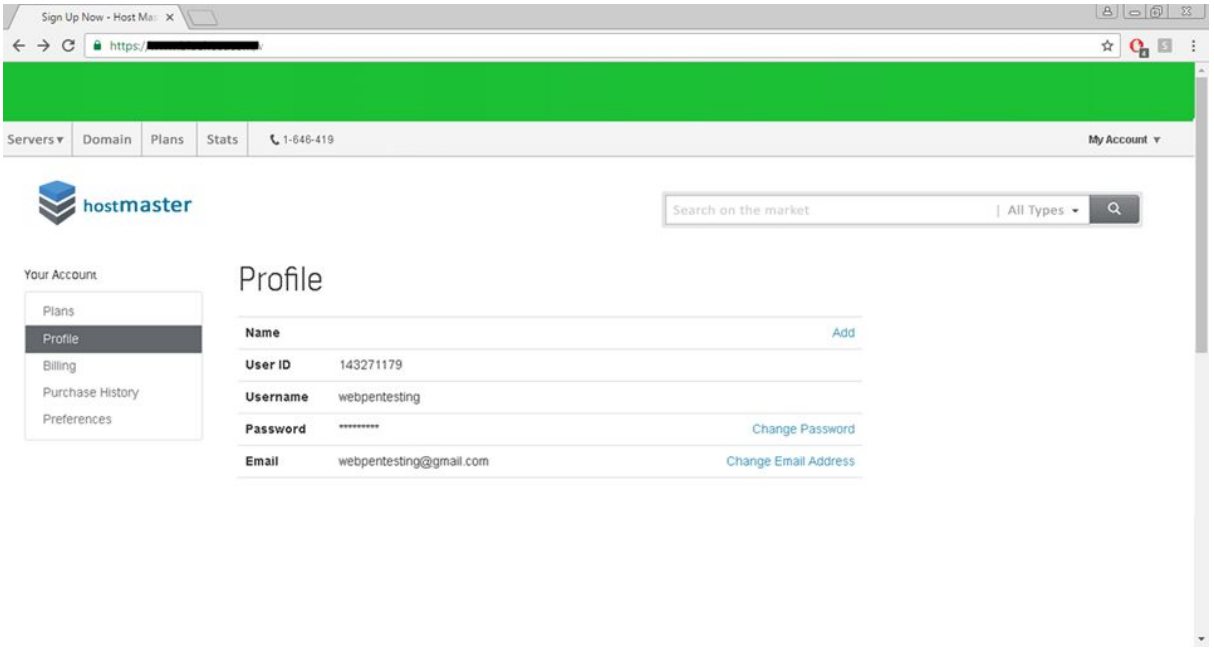
The client wanted to launch the new application in a month after the penetration test was done.

The security team discovered that the client could have launched the product with a critical vulnerability, that may have allowed any hacker to take complete control over the admin account.

Server hosting management application

The application was built for hosting companies to manage their servers and resources and it allowed users to register accounts and buy server hosting solutions.





Classic cyber attacks had no impact

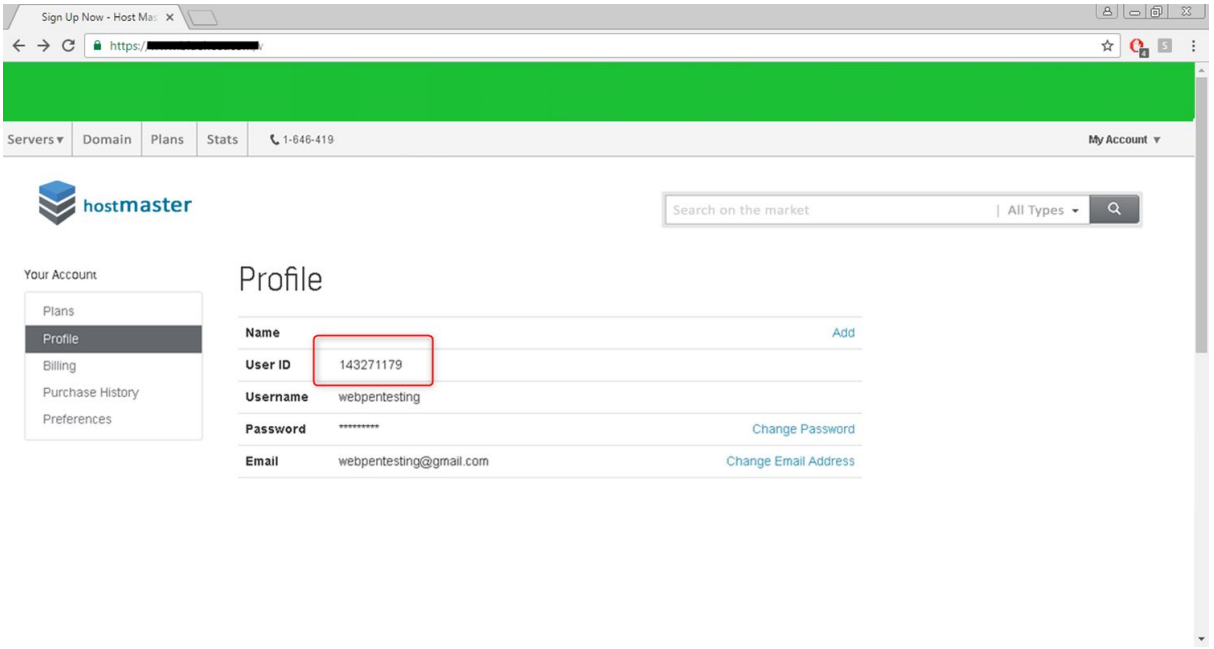
The security team attempted to penetrate the application using attacks specific to web applications, such as: SQL injection, XSS, XXE, Missing Authorizations, user brute-force, input validations and so on but they were unsuccessful due to the fact that the application was built with security in mind since the beginning.

This happens very rarely so congrats to the development team!

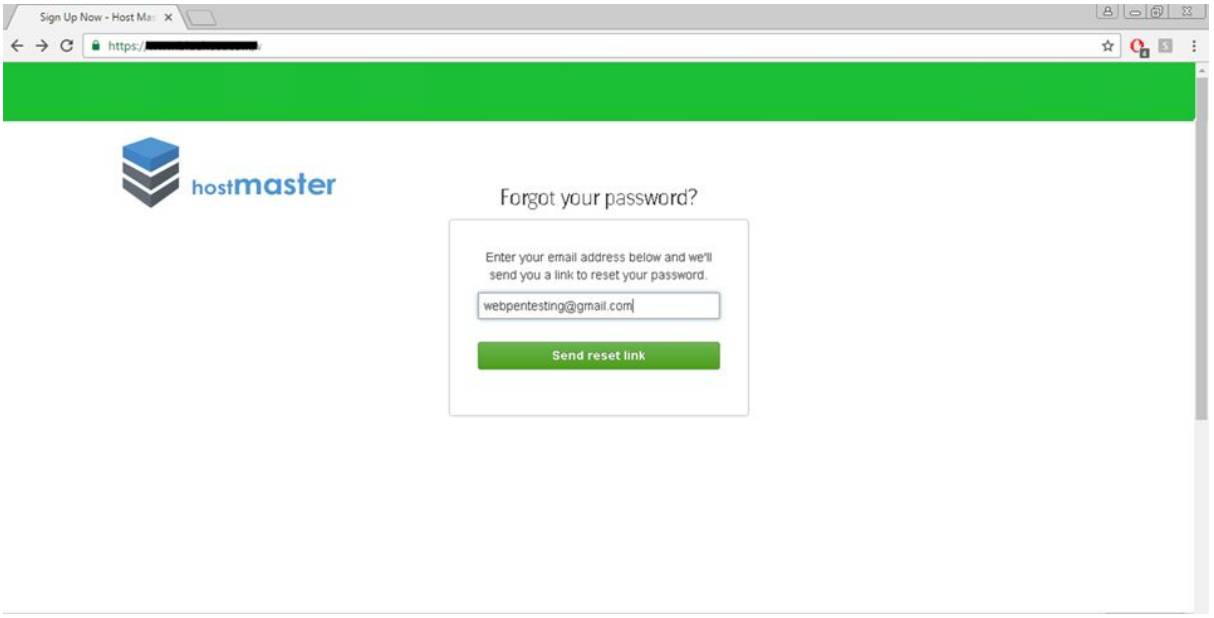
Programming error

Giving the fact that the application had defenses against most common vulnerabilities, the security team started to investigate for programming and business logic flaws.

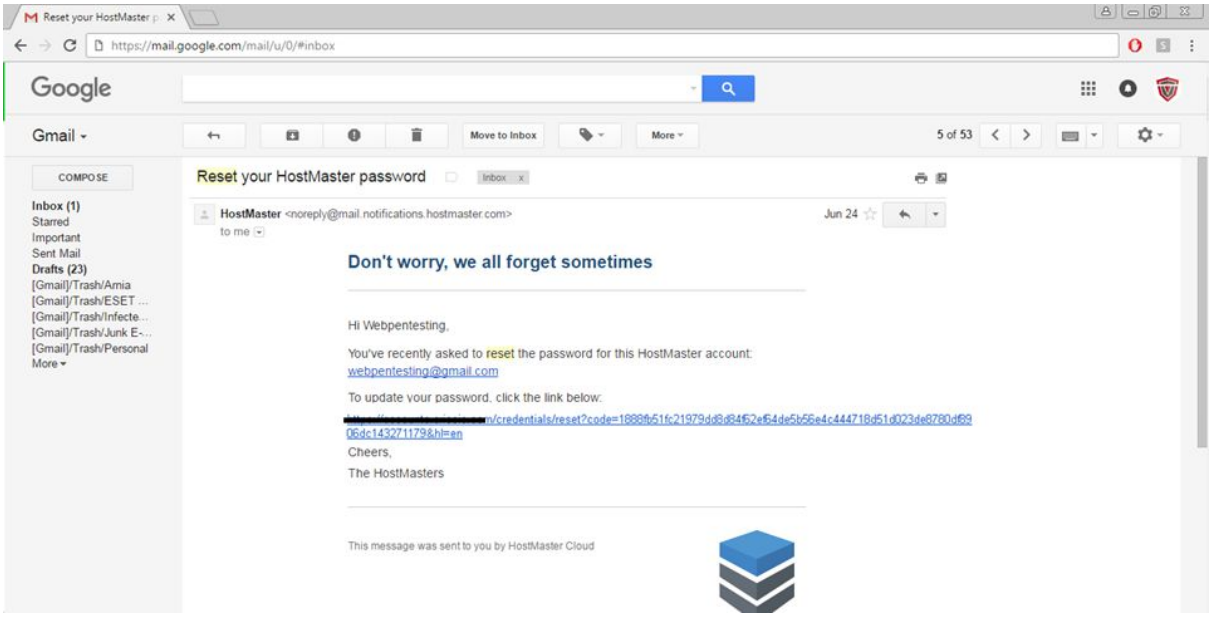
On My Account page, each user had a number assigned to their profile. Usually this number is the number that increments inside the database when a new user is registered.



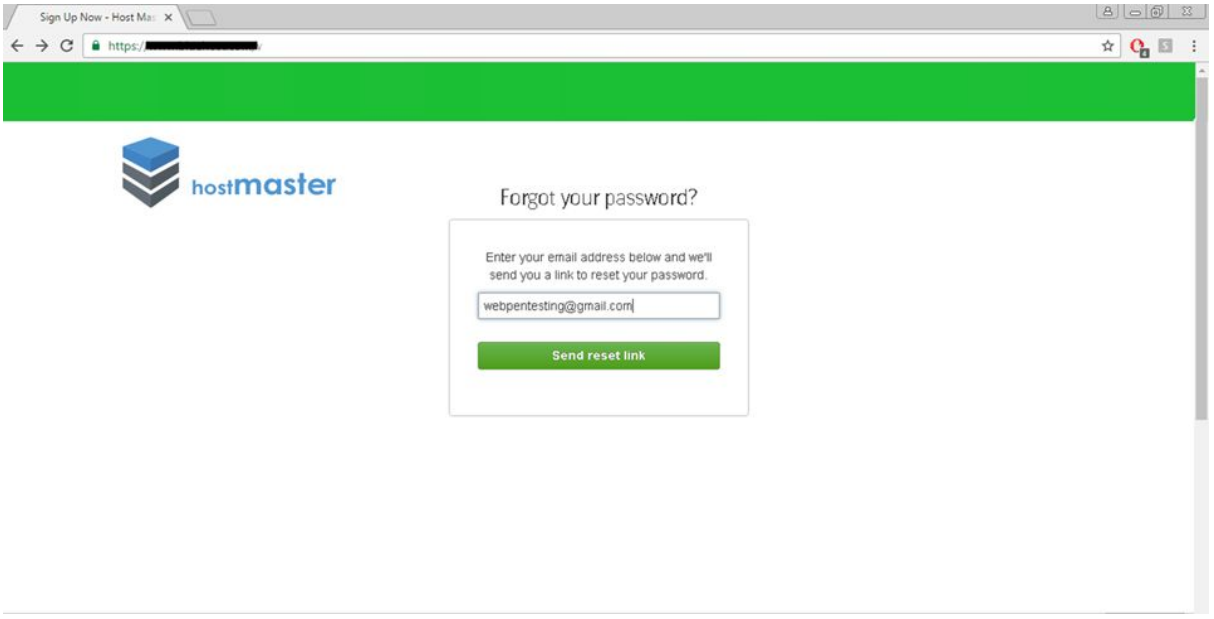
The team was looking for programming errors in **Forgot Password** section and requested a password reset mail.



It received the email with the link.



In order to compare multiple reset links, the team requested multiple Forgot Password mails on the same address.

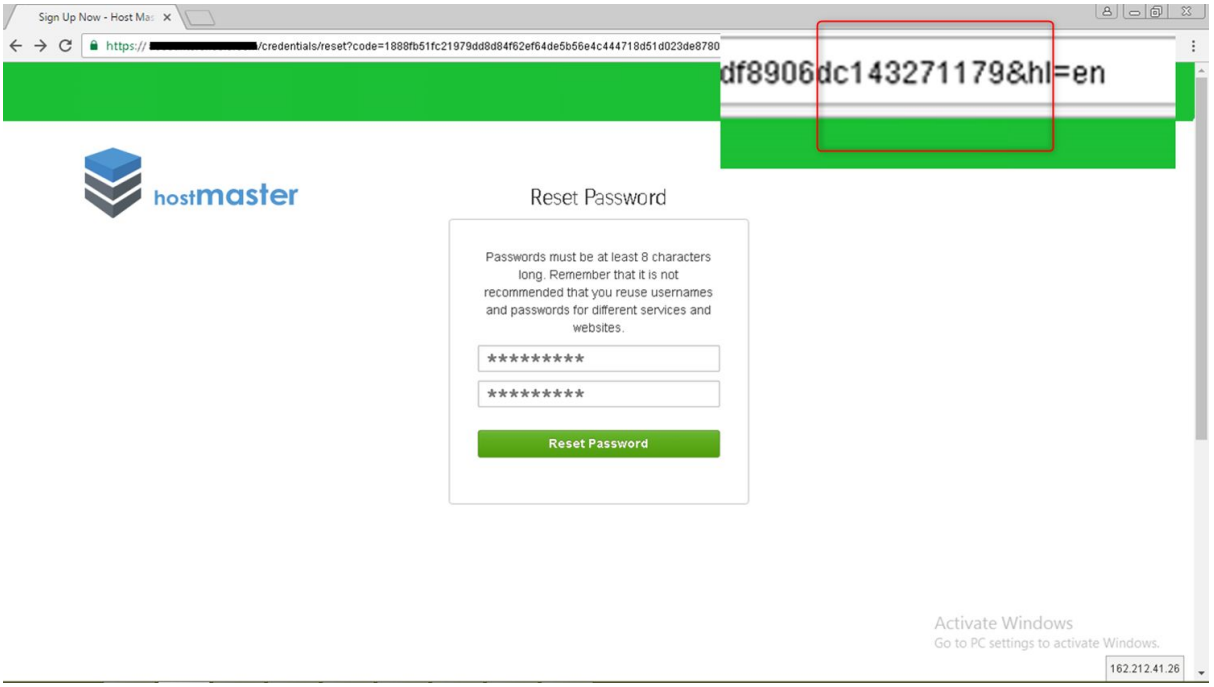


Looking at the reset password links the team observed a small coincidence. All reset links had a portion of the code identical and it was discovered that it was actually the ID of the user that requests the reset.

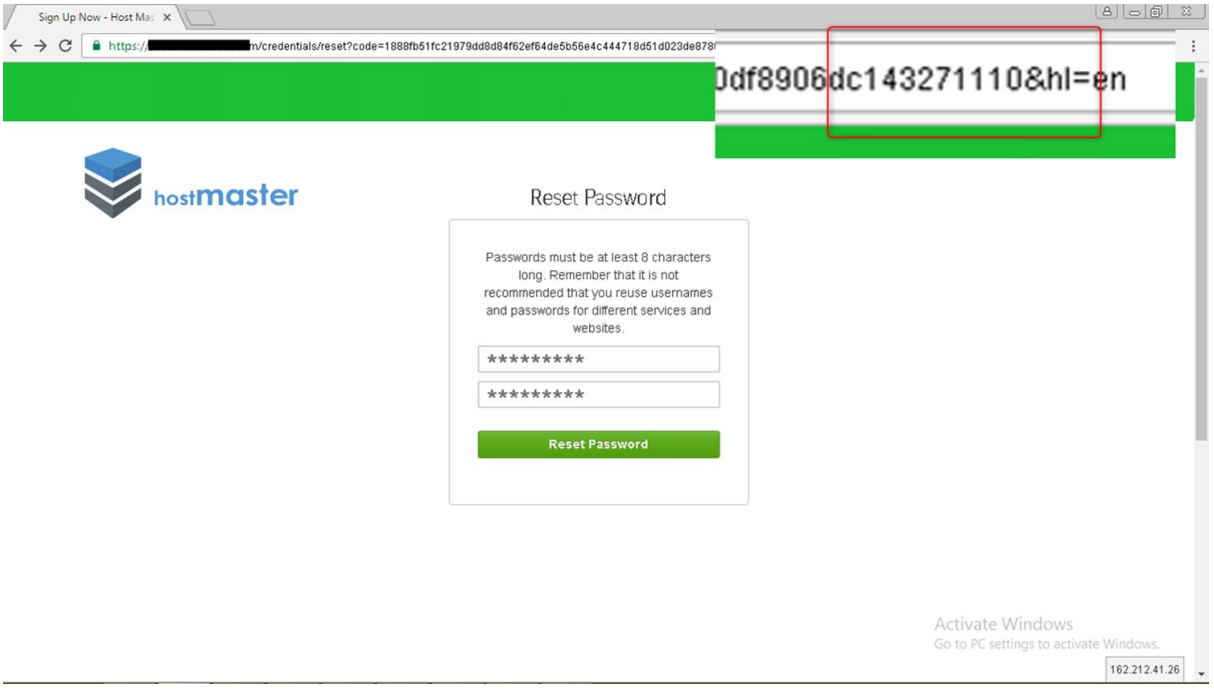
```
code=1888fb51fc21979dd8d84f62ef64de5b56e4c444718d51d023de8780df8906dc143271179&hl=en
code=316a963129374de9d8cc108c0b4ea464316a963129374de9d8cc108c0b4ea464143271179&hl=en
code=144d9dbb60b6c2c24922cd62d249412f944d9dbb60b6c2c24922cd62d249412f143271179&hl=en
code=94b9796a888b4cd1a2c1322d73999f5294b9796a888b4cd1a2c1322d73999f52143271179&hl=en
code=3f7103488be6cfcd1328ee12e40cdda63f7103488be6cfcd1328ee12e40cdda6143271179&hl=en
code=557faf8e54ec20deace42de8868e0e4e557faf8e54ec20deace42de8868e0e4e143271179&hl=en
code=32bbf0c829ec5b7f780cd5231abbf91632bbf0c829ec5b7f780cd5231abbf916143271179&hl=en
code=2c49353669a4b803a493b0677f8045ca2c49353669a4b803a493b0677f8045ca143271179&hl=en
```

```
code=1888fb51fc21979dd8d84f62ef64de5b56e4c444718d51d023de8780df8906dc143271179&hl=en
code=316a963129374de9d8cc108c0b4ea464316a963129374de9d8cc108c0b4ea464143271179&hl=en
code=144d9dbb60b6c2c24922cd62d249412f944d9dbb60b6c2c24922cd62d249412f143271179&hl=en
code=94b9796a888b4cd1a2c1322d73999f5294b9796a888b4cd1a2c1322d73999f52143271179&hl=en
code=3f7103488be6cfcd1328ee12e40cdda63f7103488be6cfcd1328ee12e40cdda6143271179&hl=en
code=557faf8e54ec20deace42de8868e0e4e557faf8e54ec20deace42de8868e0e4e143271179&hl=en
code=32bbf0c829ec5b7f780cd5231abbf91632bbf0c829ec5b7f780cd5231abbf916143271179&hl=en
code=2c49353669a4b803a493b0677f8045ca2c49353669a4b803a493b0677f8045ca143271179&hl=en
```

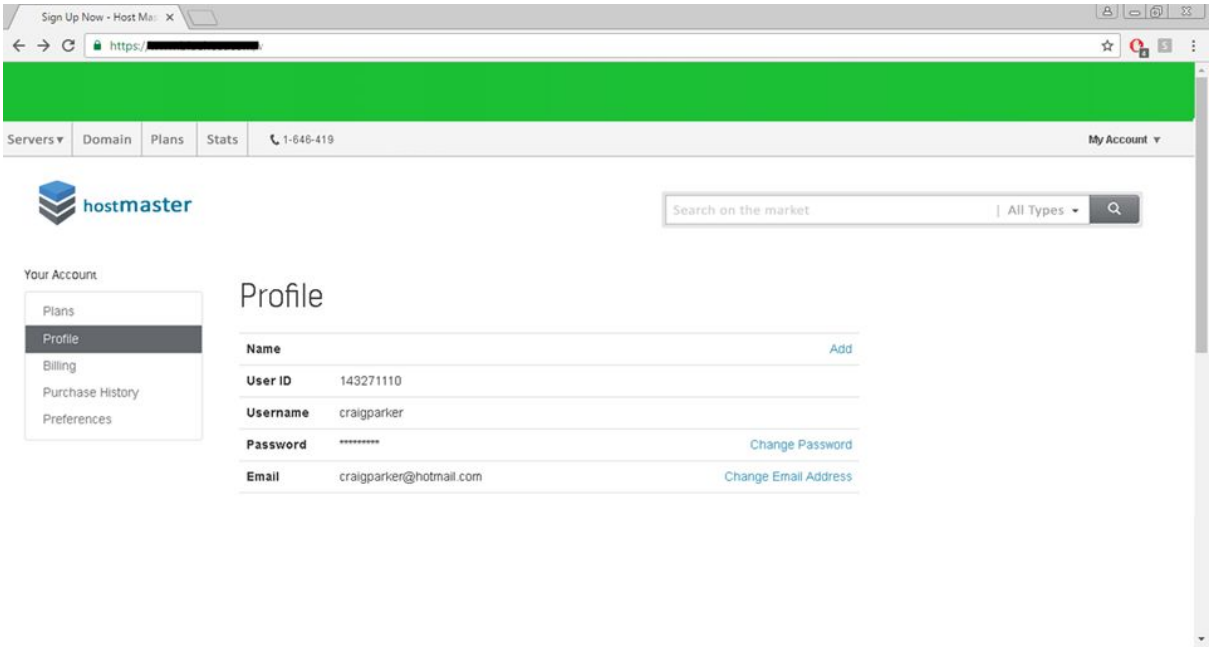
The next step was to access the link in the URL and change the ID of the user with other user's IDs, by incrementing its value.



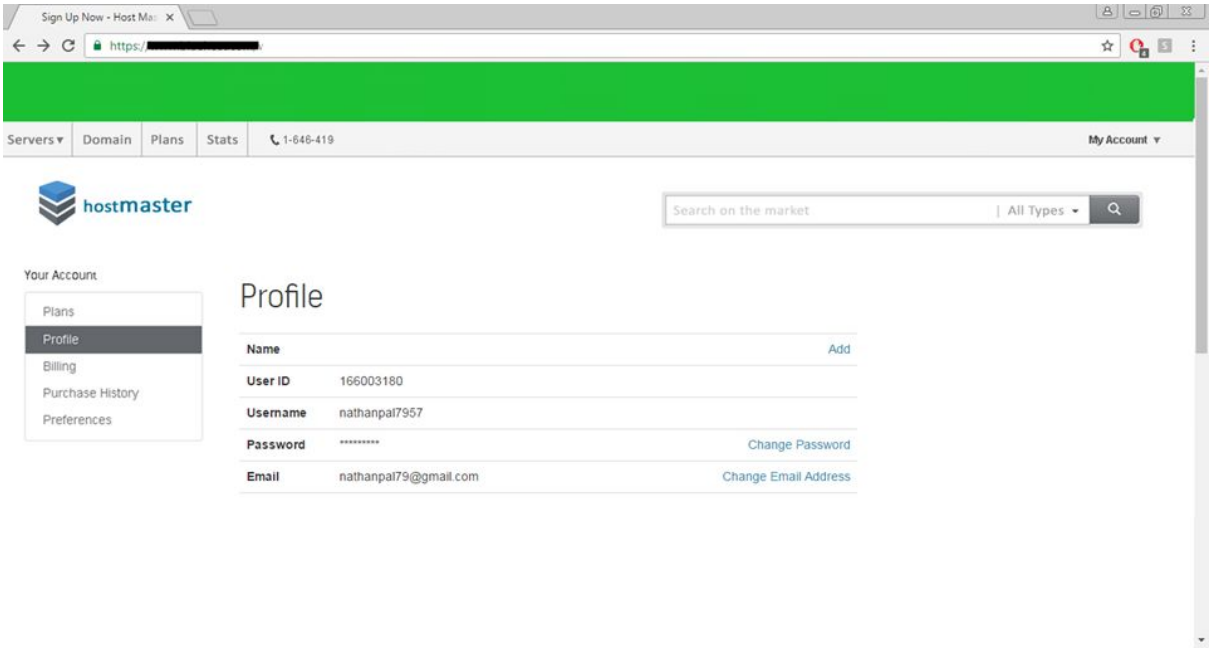
Other user ID.



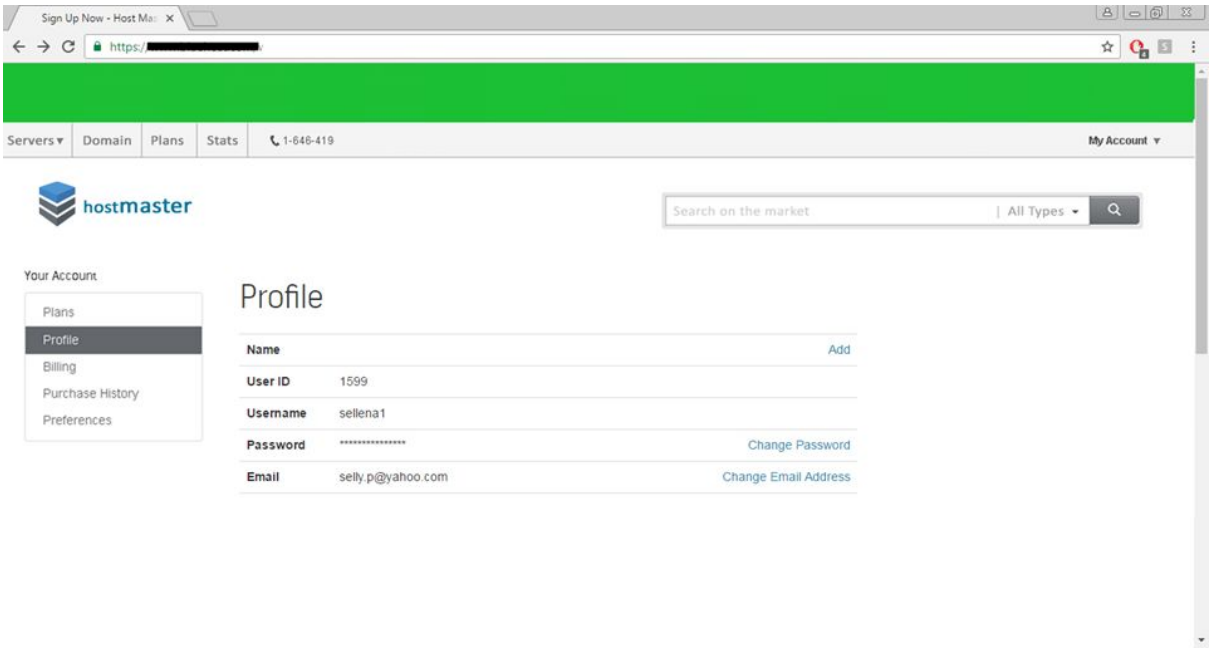
Once the request was sent the application redirected to the Account page of each user where the password was reset.



And other



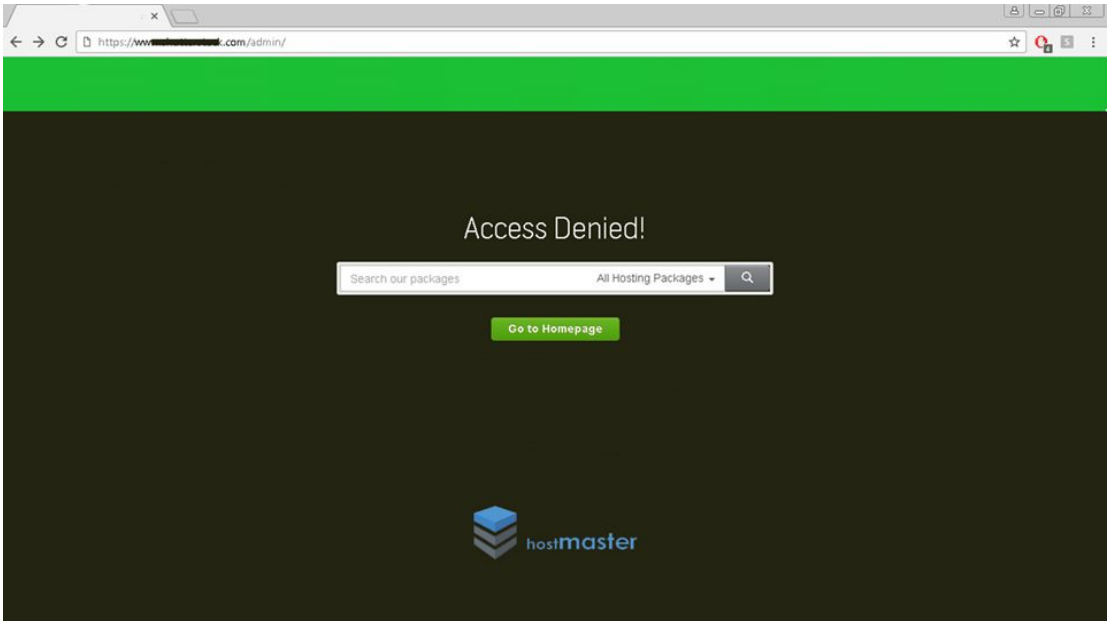
And other



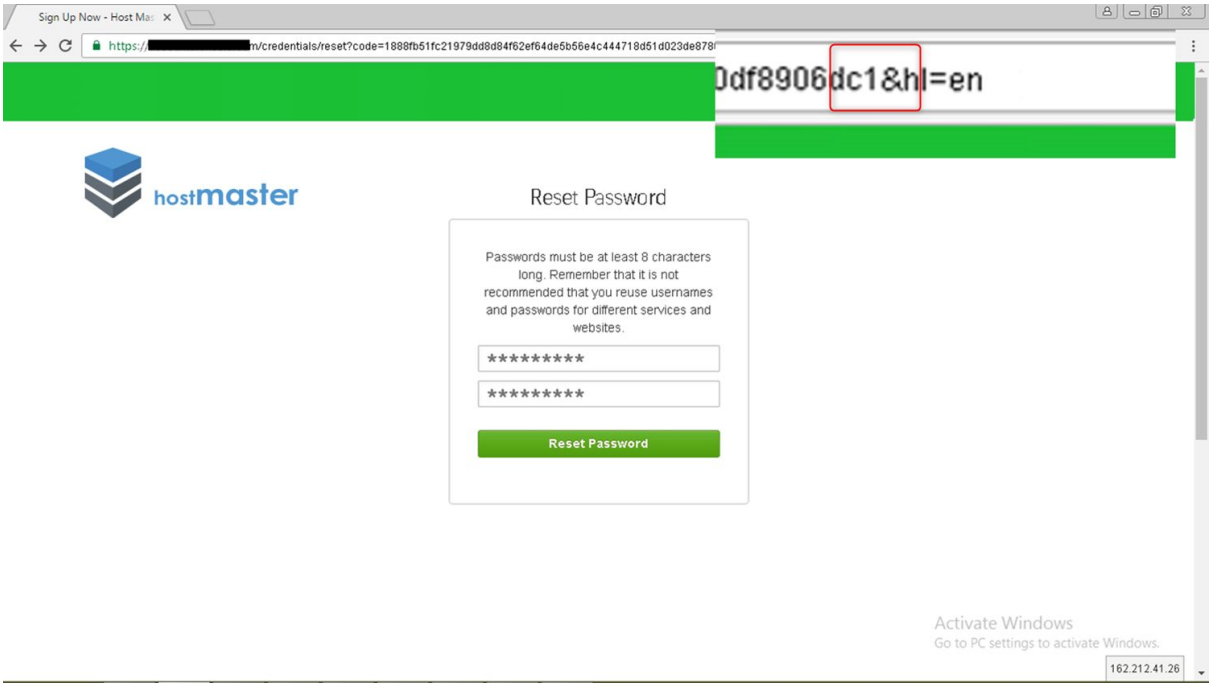
In this manner the team managed to change the passwords of all users in the system and obtain access to all accounts.

Admin page access

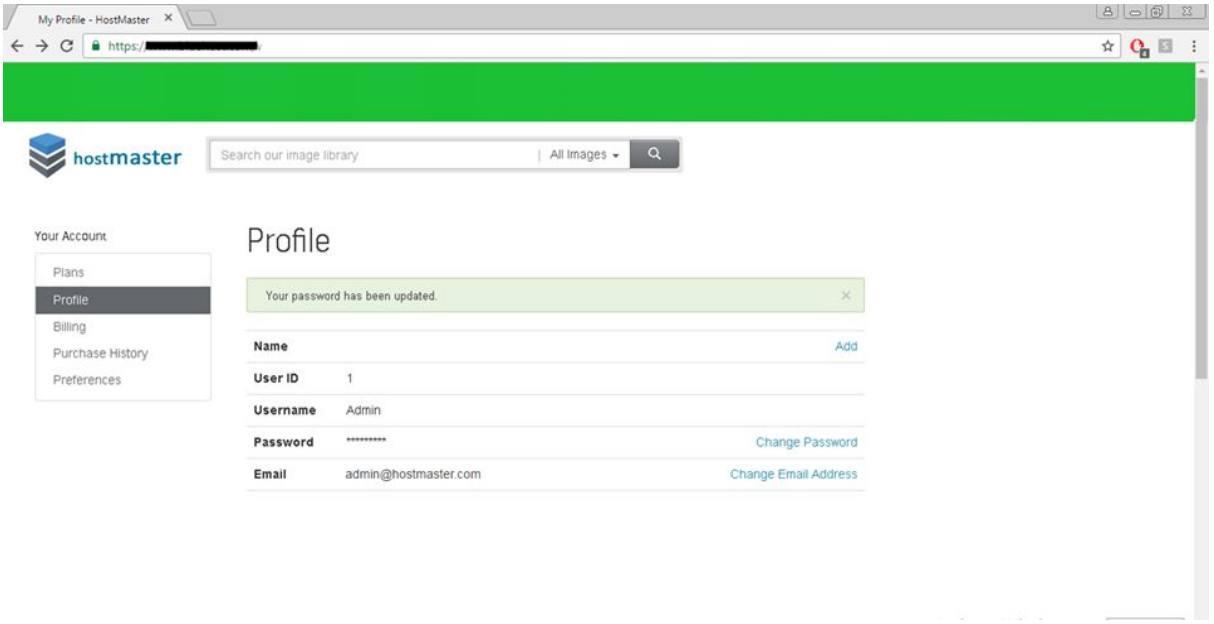
The next step was to try to access the admin panel, but it looked like the team didn't have enough permissions with any of the users.



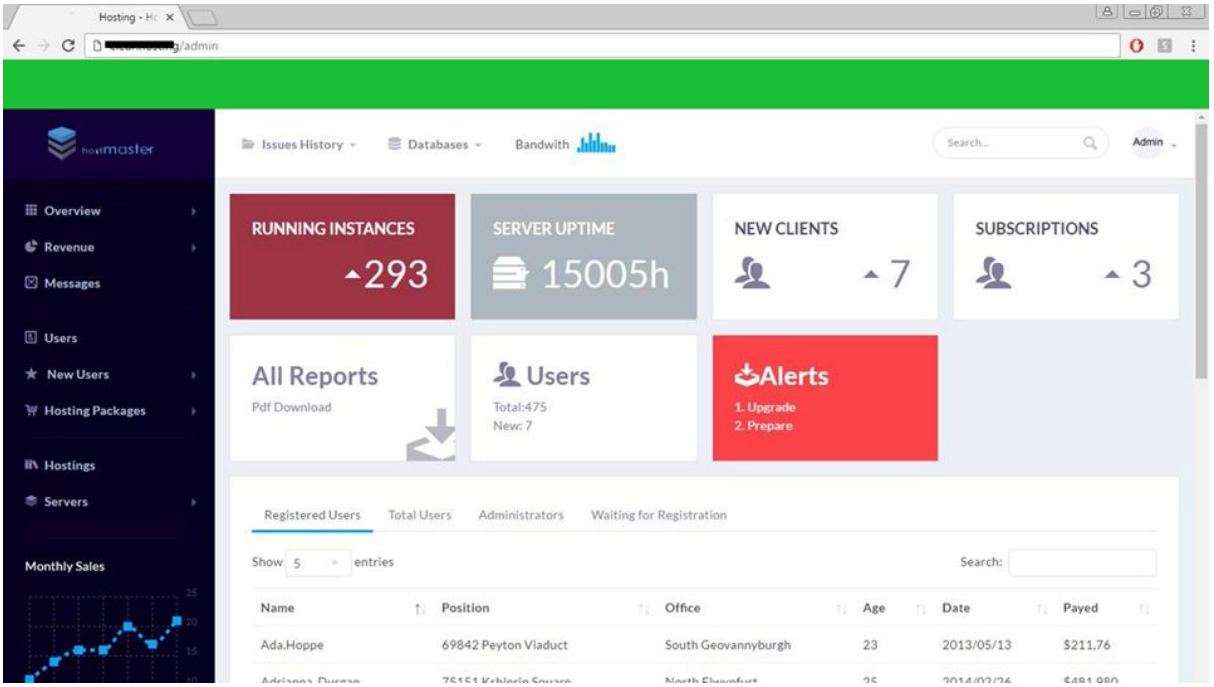
Coming back to the new vulnerability detected the team tried to reset the password of the user with the id=1. Why? When a new application is developed usually the first user created is the admin and has ID 1 in the database. If this doesn't work, then it could be -1.



And what was discovered is that the team managed to reset the password for the admin account.



Next step was to access the admin page once again.



The team had complete access over the application.

Conclusion

Even though the application is build with security requirements, an experienced penetration tester, with a background in development can find business logic vulnerabilities that can still produce a lot of risk. In this case, an attacker with complete admin access could have had the possibility to redirect payment transactions to his account, deny users to buy products, close servers of existing customers making them lose users and money.

The client was helped to mitigate the risk with support from the security team. Also, the client decided to include a penetration tester in the requirements and implementation phase of next web application developed.