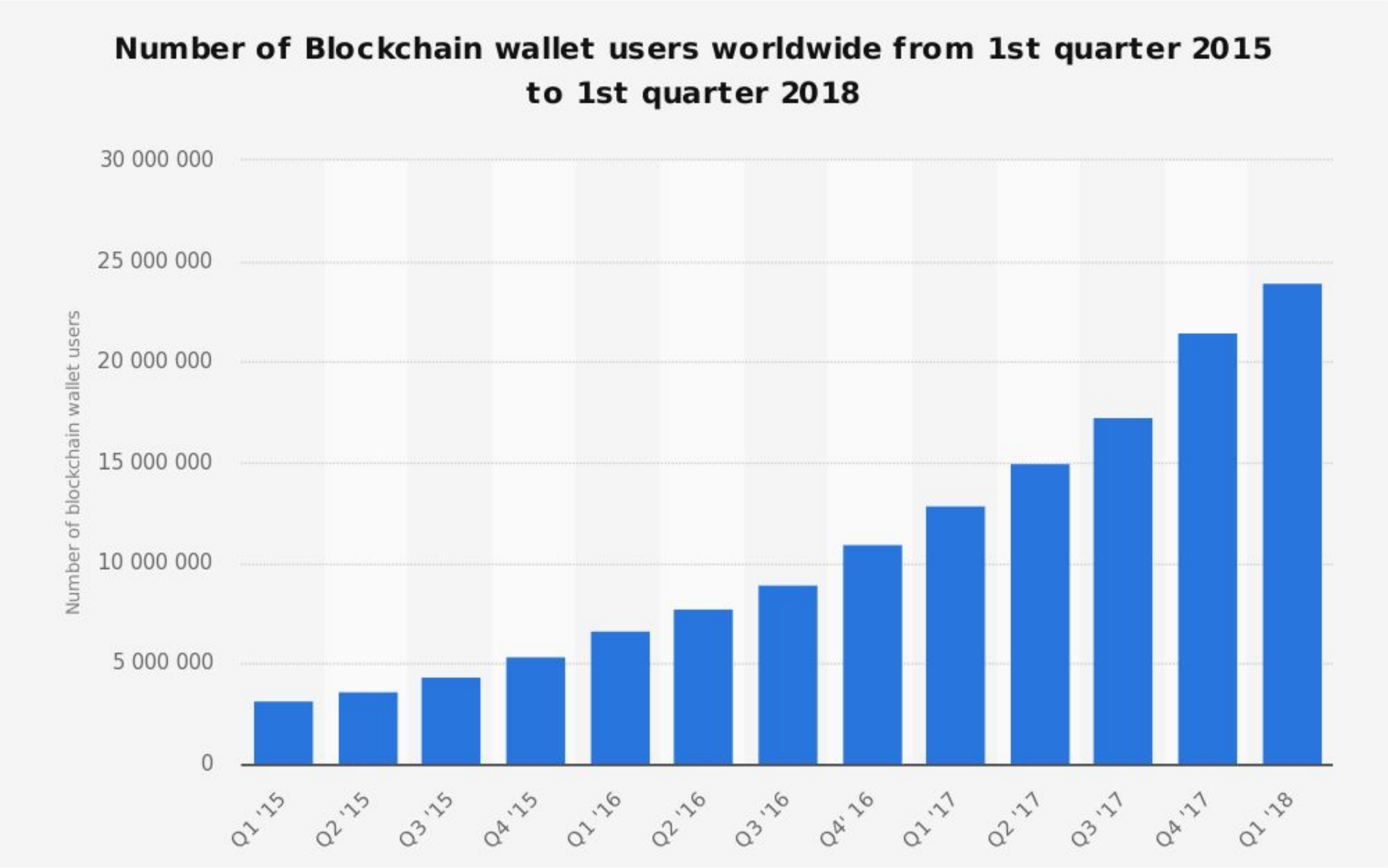
Case Study

UNDER
DEFENSE
CyberSecurity Solutions
Protecting your business

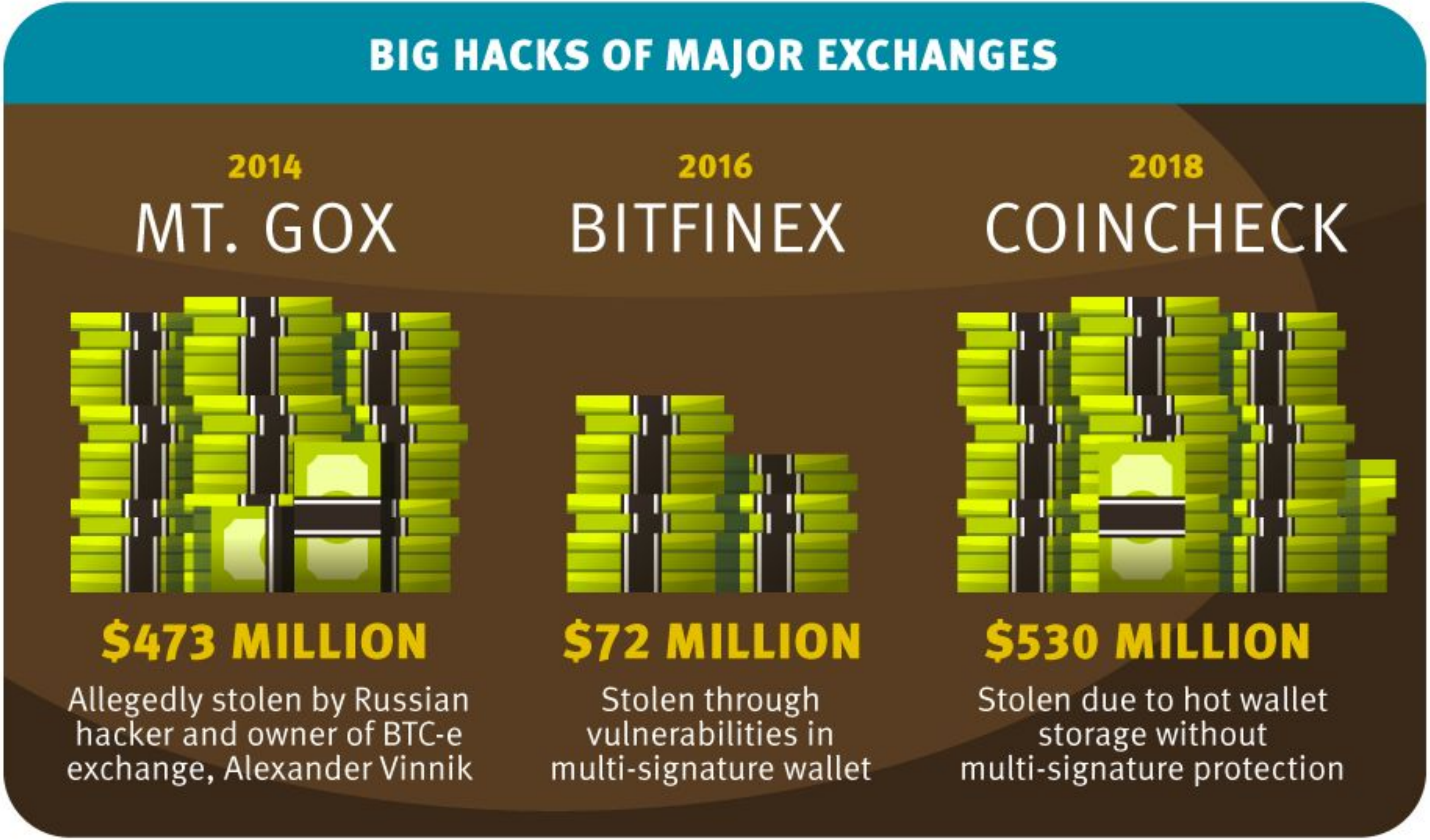
Cryptocurrency marketplace and smart-contract Security

Black Box Security Assessment

Solution/Service Title	Security Testing for Platform for money transfer operations, Cryptocurrency marketplace, Solidity based smart-contract security audit
Client Industry	Blockchain and bitcoins money transfering and exchanging
Client Overview	A Hong Kong based Fintech company that provides a secure web and mobile platform for money transfer operators, send and receive cash remittance payments to over 100,000 cash out locations globally.
Client Challenge	Recent data breaches of popular cryptocurrency marketplaces and bitcoin wallets made a client interested in security testing of financial transactions and platform security to avoid security issues in the future and ensure safety of clients data
Scope	Solidity based Smart-contract, IT infrastructure, Web Application, subdomains, API, Mobile application
Key Benefits	Independent security audit/review allowed our client to avoid data breach, improve IT/Security processes and follow best practices.
Results	Discovered critical and high issues could lead to full application compromise, unauthorized financial transaction and lost of clients money, reputation and trust.



Number of Blockchain wallet users worldwide from 1st quarter 2015 to 1st quarter 2018



BIG HACKS OF MAJOR EXCHANGES

2014

MT. GOX

$473 MILLION

Allegedly stolen by Russian hacker and owner of BTC-e exchange, Alexander Vinnik

2016

BITFINEX

$72 MILLION

Stolen through vulnerabilities in multi-signature wallet

2018

COINCHECK

$530 MILLION

Stolen due to hot wallet storage without multi-signature protection

Project Background

Client

- A Hong Kong based fintech company

Technical Goal

- Find what can do an attacker without initial knowledge of a target (black-box ethical hacking). Detect and fix security issues to save sensitive data and money

Business Goals

- Evaluate current level of business and platform security, identify gaps in current cyber security program, check IT environment and smart-contract for weaknesses.

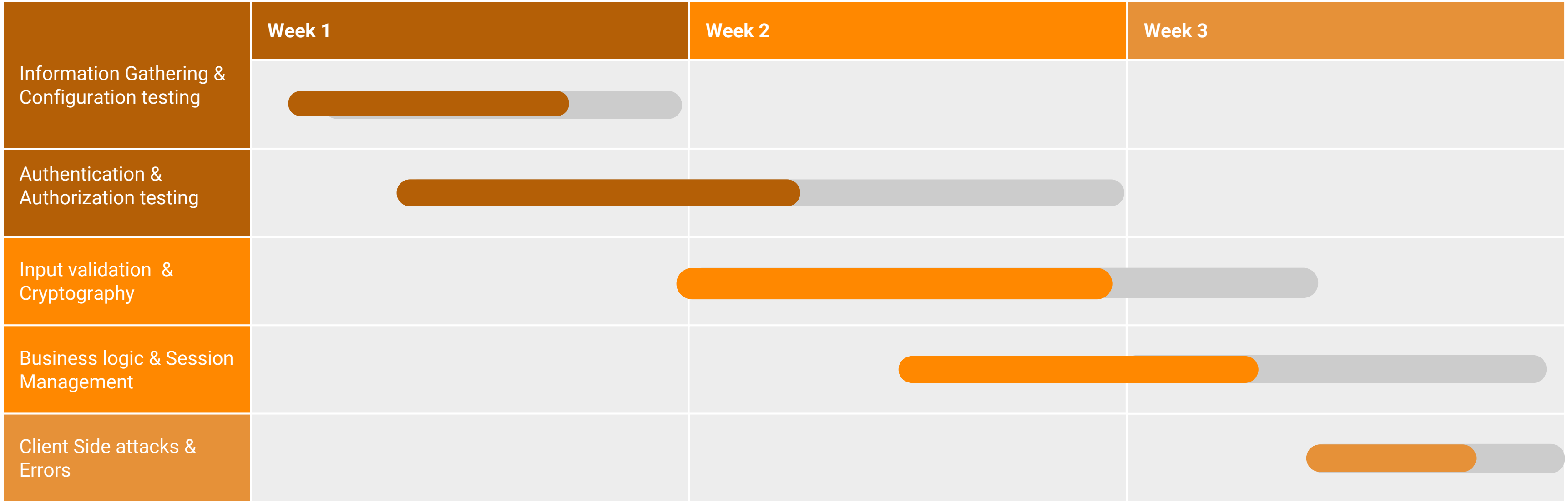
Team

- 2 certified ethical hackers

Duration

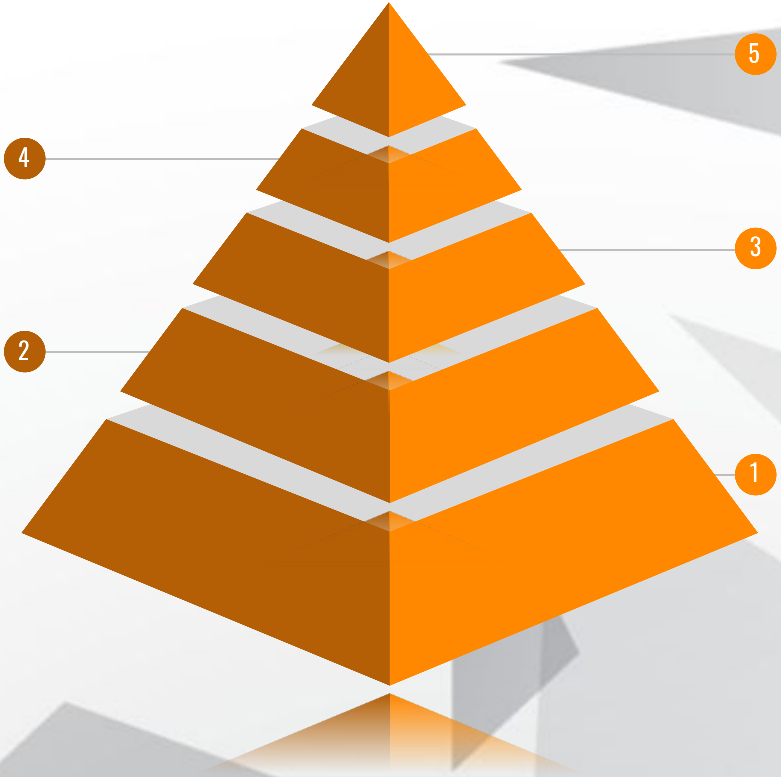
- 3 weeks

Project planning and goals



	Week 1	Week 2	Week 3
Information Gathering & Configuration testing			
Authentication & Authorization testing			
Input validation & Cryptography			
Business logic & Session Management			
Client Side attacks & Errors			

Attack Vectors



Get access to money

5 Get the ability to operate other users’ money.

Get access to admin account

4 Get access to administrative functions and data.

Get access to user account

3 Get access to user’s data and available functions.

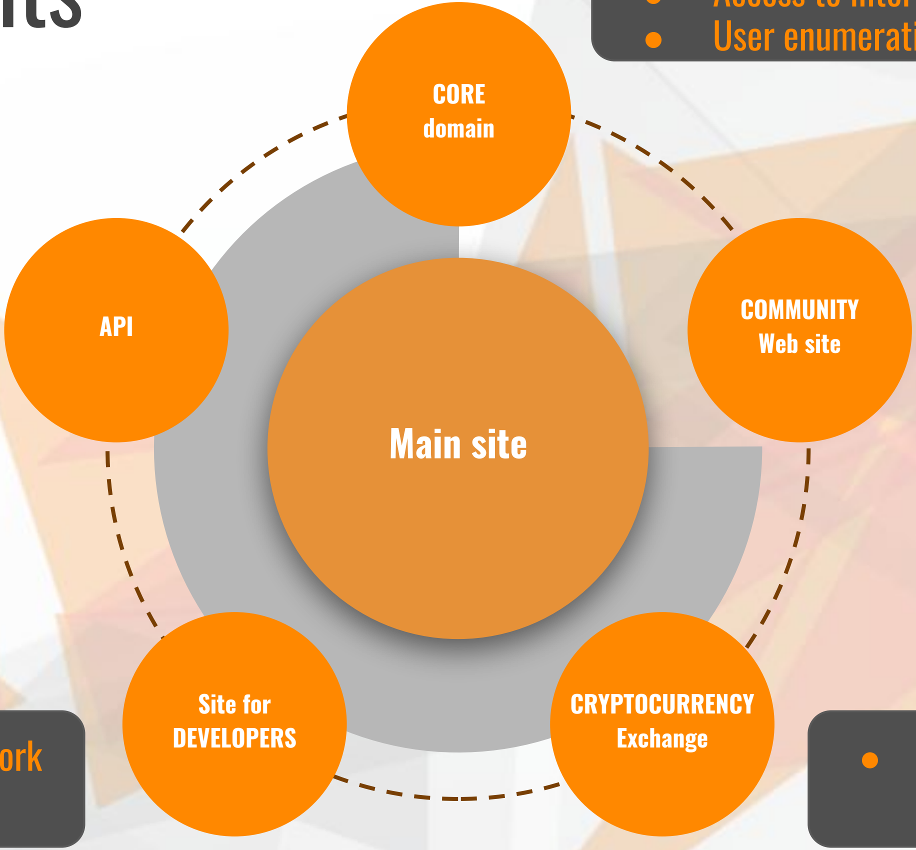
Find privileged users

2 Get list of administrators of the system or users that might have some privileges.

Find valid users

1 Use different techniques to enumerate valid users of the system.

Inception points



- User register
- Access to internal functions
- User enumeration

- Access to API functions

- Information about real users and possible admins
- Additional info about service

- Deeply explained how to work with API

- Old community open source version on github

Key moments

We had

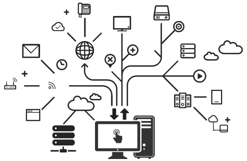
- No knowledge and information about the application and infrastructure
- No usual or privileged accounts
- ONE name of the target

We have got

- Information about the application architecture
- List of valid users
- List of valid admins
- Access to users and admin account
- Unauthorized access to users data and money

Technologies and tools used

20%

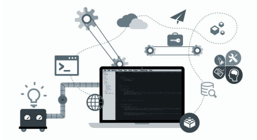


Infrastructure analysis

The beginning of each security check usually starts with the examination of the application surround, which is an infrastructure. Tools we had used for this purpose made some automating checks to discover the architecture and relations between applications.

- ➢ Nmap
- ➢ TestSSL
- ➢ Dig
- ➢ Nslookup
- ➢ Nessus
- ➢ Nexpose

70%



Web Application Analysis

Main focus was on the analysis of the web application as it was a core part of whole infrastructure. Security tools we used for WEB application testing helped us to automate some routine work and accelerate the process of pentesting.

- ➢ Burp Suite
- ➢ Dirbuster
- ➢ Nikto
- ➢ Tachyon

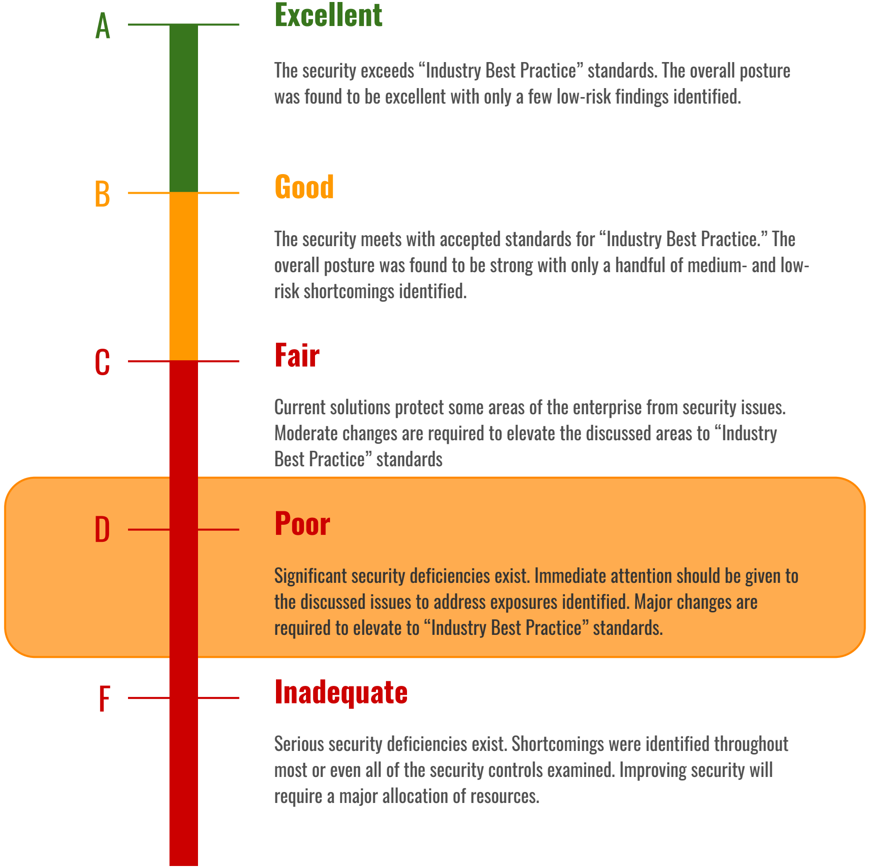
10%

Mobile Application Analysis

Mobile applications usually duplicates the WEB application functionality, but still contains more weaknesses in design and implementation. Tools we used provide us with the details of static code analysis and makes dynamic analysis easier.

- ➢ MobFS
- ➢ Inspeckage
- ➢ Xposed Framework

Current Product Security Level Evaluation State:



Excellent

The security exceeds “Industry Best Practice” standards. The overall posture was found to be excellent with only a few low-risk findings identified.

Good

The security meets with accepted standards for “Industry Best Practice.” The overall posture was found to be strong with only a handful of medium- and low-risk shortcomings identified.

Fair

Current solutions protect some areas of the enterprise from security issues. Moderate changes are required to elevate the discussed areas to “Industry Best Practice” standards

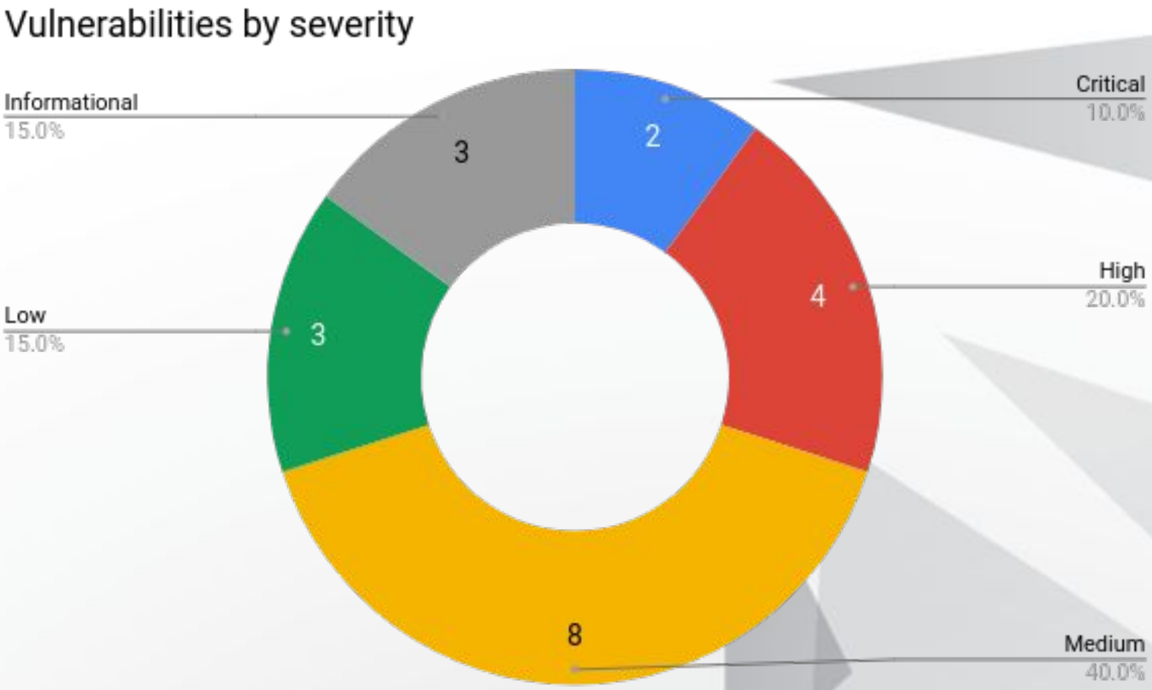
Poor

Significant security deficiencies exist. Immediate attention should be given to the discussed issues to address exposures identified. Major changes are required to elevate to “Industry Best Practice” standards.

Inadequate

Serious security deficiencies exist. Shortcomings were identified throughout most or even all of the security controls examined. Improving security will require a major allocation of resources.

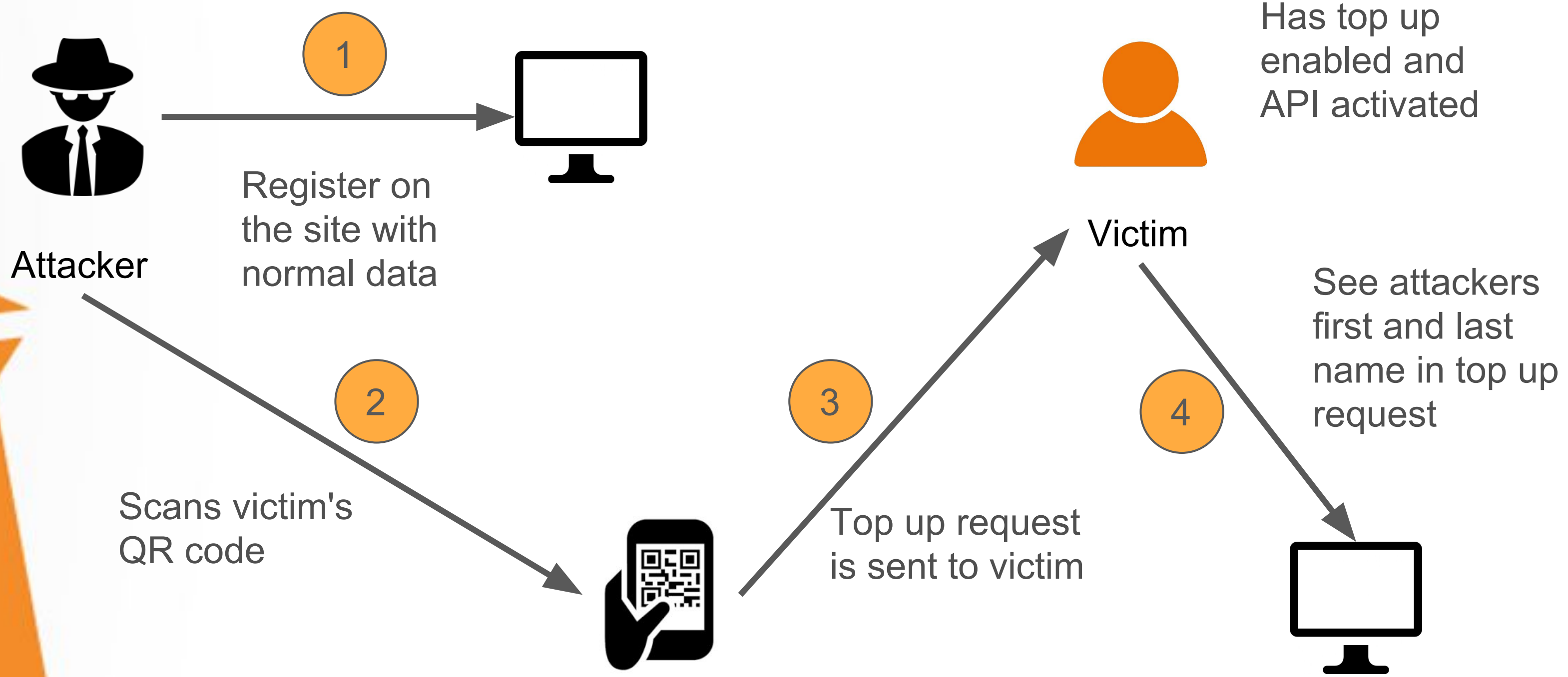
Security issues



OWASP Top 10 Security Threats 2017

A1:2017	Injection	Meets Criteria
A2:2017	Broken Authentication	Fails Criteria
A3:2017	Sensitive Data Exposure	Fails Criteria
A4:2017	XML External Entities (XXE)	Meets Criteria
A5:2017	Broken Access Control	Meets Criteria
A6:2017	Security Misconfiguration	Fails Criteria
A7:2017	Cross Site Scripting (XSS)	Fails Criteria
A8:2017	Insecure Deserialization	Meets Criteria
A9:2017	Using Components With Known Vulnerabilities	Fails Criteria
A10:2017	Insufficient Logging & Monitoring	Fails Criteria

Critical Issues - Stored XSS in Top_Up



Register on the site with normal data

Scans victim's QR code

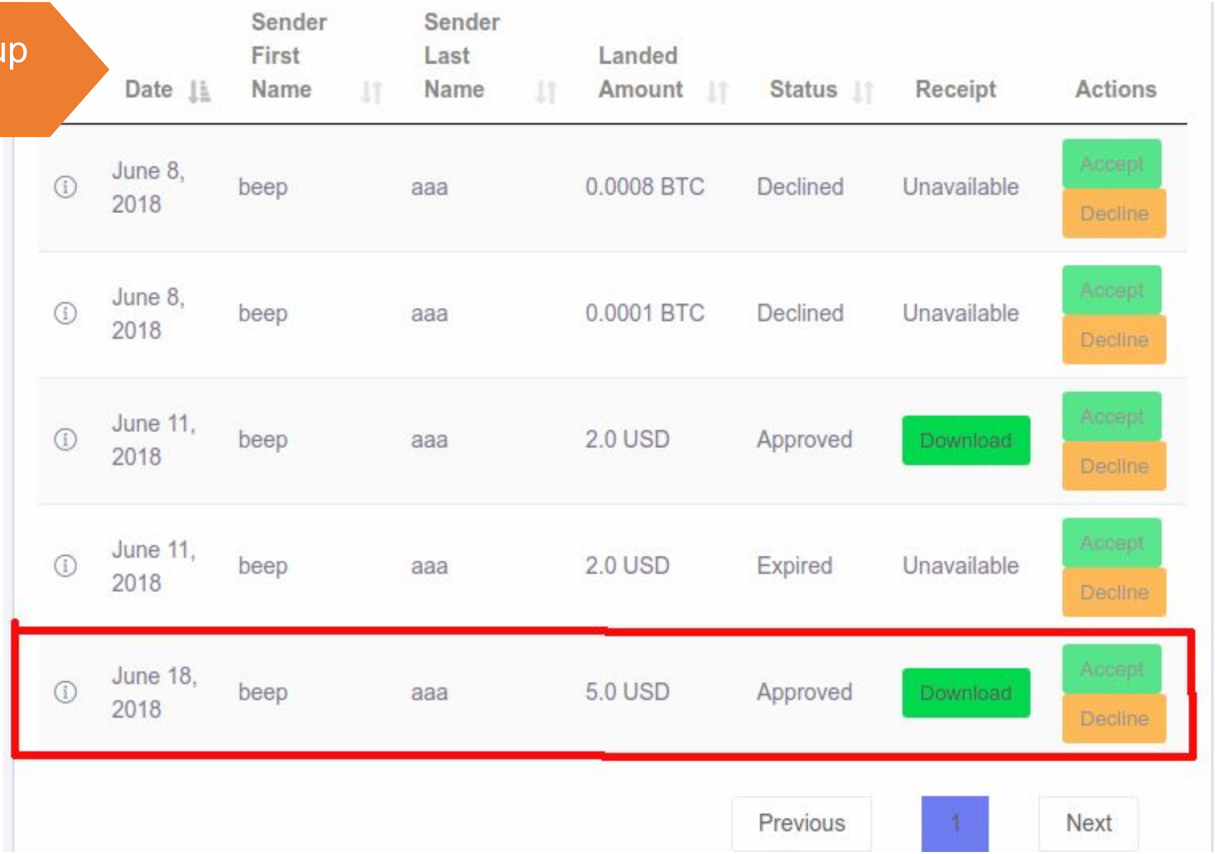
Top up request is sent to victim

Has top up enabled and API activated

See attackers first and last name in top up request

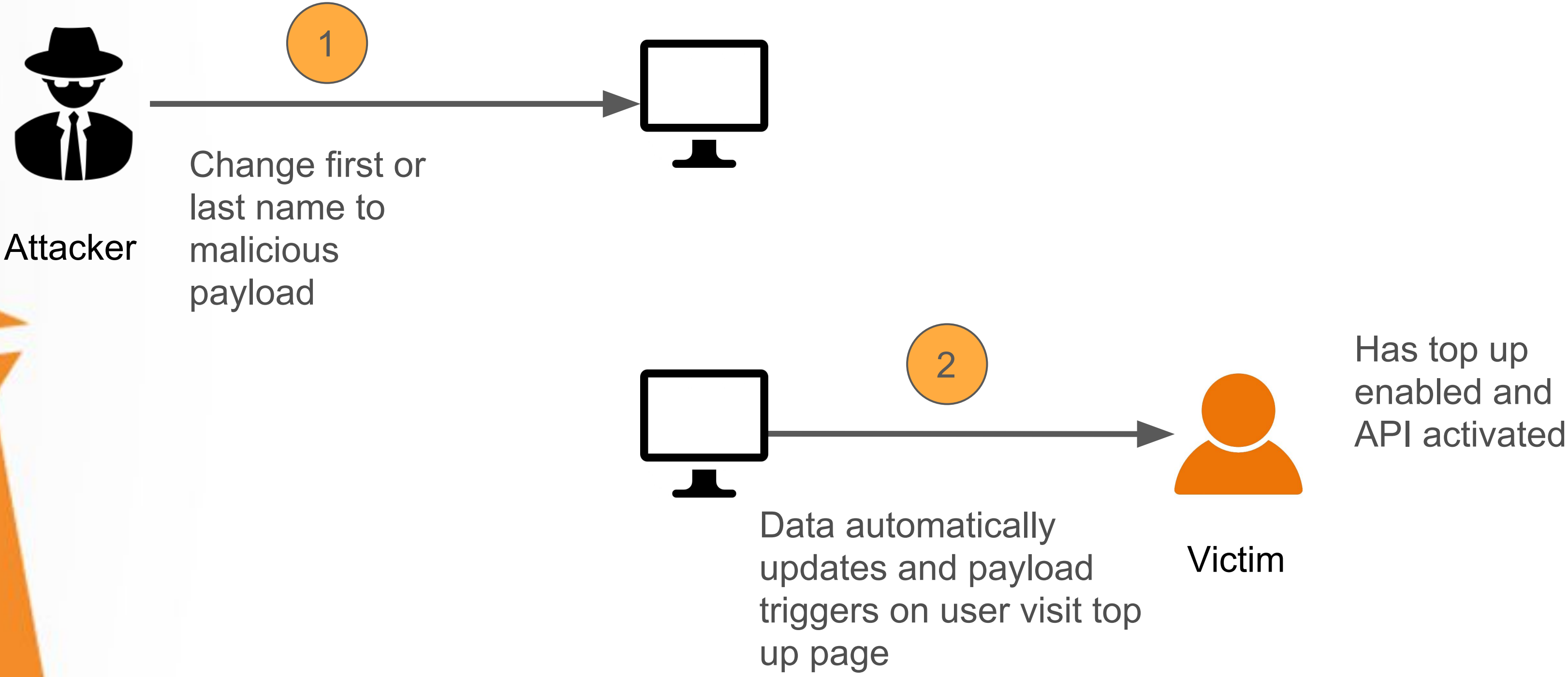
Critical Issues - Stored XSS in Top_Up

Victims view of top up page



	Date	Sender First Name	Sender Last Name	Landed Amount	Status	Receipt	Actions
ⓘ	June 8, 2018	beep	aaa	0.0008 BTC	Declined	Unavailable	Accept Decline
ⓘ	June 8, 2018	beep	aaa	0.0001 BTC	Declined	Unavailable	Accept Decline
ⓘ	June 11, 2018	beep	aaa	2.0 USD	Approved	Download	Accept Decline
ⓘ	June 11, 2018	beep	aaa	2.0 USD	Expired	Unavailable	Accept Decline
ⓘ	June 18, 2018	beep	aaa	5.0 USD	Approved	Download	Accept Decline

Critical Issues - Stored XSS in Top_Up



Change first or last name to malicious payload

Data automatically updates and payload triggers on user visit top up page

Has top up enabled and API activated

High Issues - Redis Server Unprotected by Password Authentication

```
#                                                     Server
redis_version:3.2.11
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:6f45701c6c1a40a0
redis_mode:standalone
os:Linux               4.4.0-119-generic              x86_64
arch_bits:64
multiplexing_api:epoll
gcc_version:4.9.2
process_id:1
run_id:f8c976090e4791b1b9d8501491fab17fe01ac3f3
tcp_port:6379
uptime_in_seconds:602289
uptime_in_days:6
hz:10
lru_clock:2075900
executable:/data/redis-server
config_file:
```

Business risks detected

Financial loss

Reputational damage

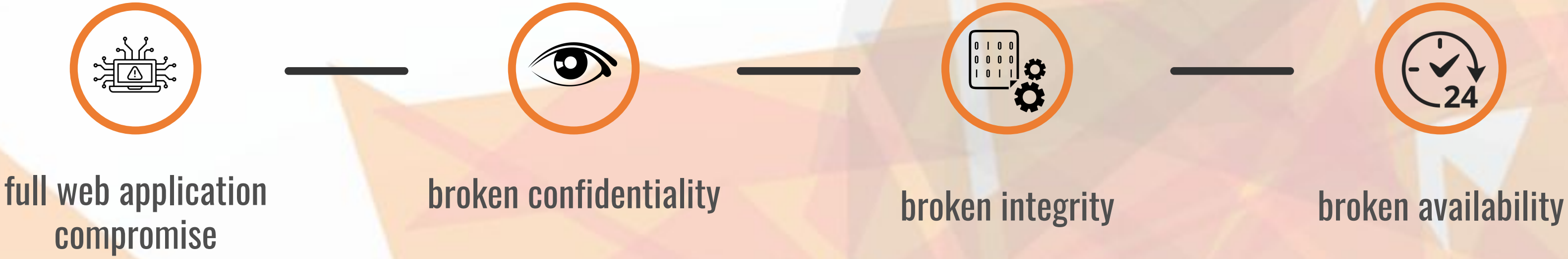
Loss of clients trust

Data loss

- Full web application compromise
- Admin account compromise
- User account compromised
- Ability to make transaction
- Ability to eavesdrop communication

Summary

The test uncovered a critical vulnerabilities that cause:



full web application compromise

broken confidentiality

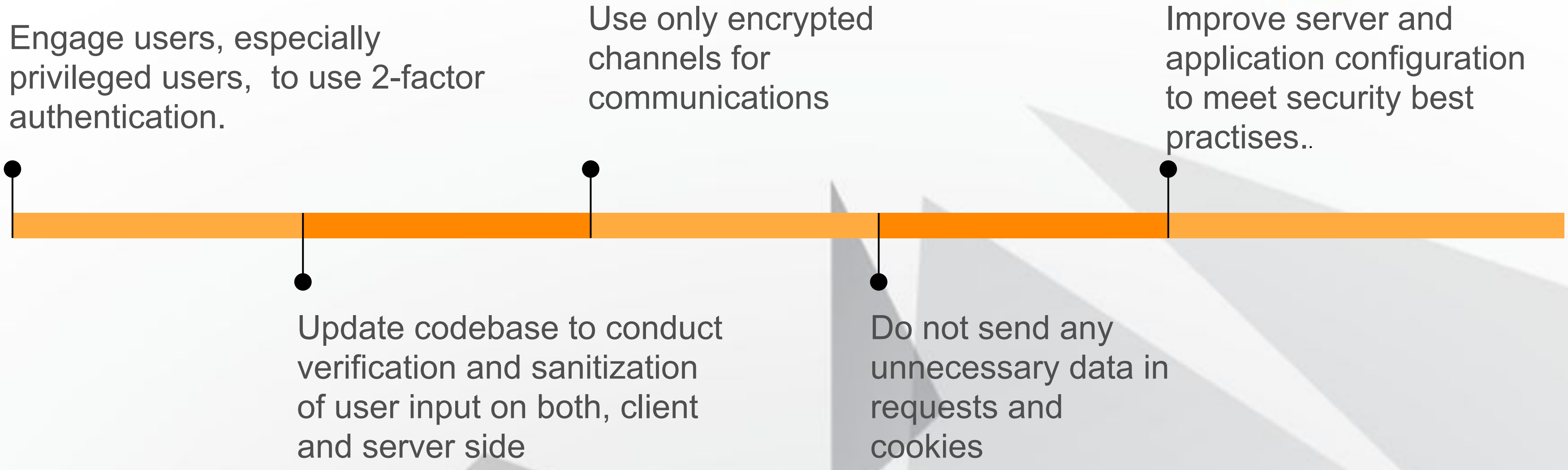
broken integrity

broken availability

Taking into consideration all findings and risks that they bring, we divided our recommendation on two parts:

1. Require immediate actions
2. Strategic recommendations, that intend to minimize security risks in the future.

Immediate recommendations



Engage users, especially privileged users, to use 2-factor authentication.

Update codebase to conduct verification and sanitization of user input on both, client and server side

Use only encrypted channels for communications

Do not send any unnecessary data in requests and cookies

Improve server and application configuration to meet security best practises..

Strategic recommendations

- Conduct current vs. future IT/Security program review
- Conduct Static code analysis for ruby codebase
- Establish Secure SDLC best practices
- Review Architecture of application
- Deploy Web Application Firewall
- Continuously monitor logs for anomalies to detect abnormal behaviour and fraud transactions

Strategic recommendations

- Implement Patch Management procedures
- Conduct annual Penetration test and quarterly Vulnerability Scanning
- Conduct security coding training for Developers
- Develop and Conduct Security Awareness training for employees and developers
- Develop Incident Response Plan in case if of Data breach or security incidents
- Analyse risks for key assets and resources

Thank you!

Call us now at +1 929 999 5101