Security
in Emergency Situations

Jun Yi, Yiming Peng

We Need Protection

Project Outline

- Login and different authorities
- Communication encrypted based on **Kerberos Algorithm**
- Database encryption and certification
- Friendly UI for server administrator
- **Security token** for administrator login
- **Three levels** of security protection
- **Security test** using different attacks

Security Levels

Purpose

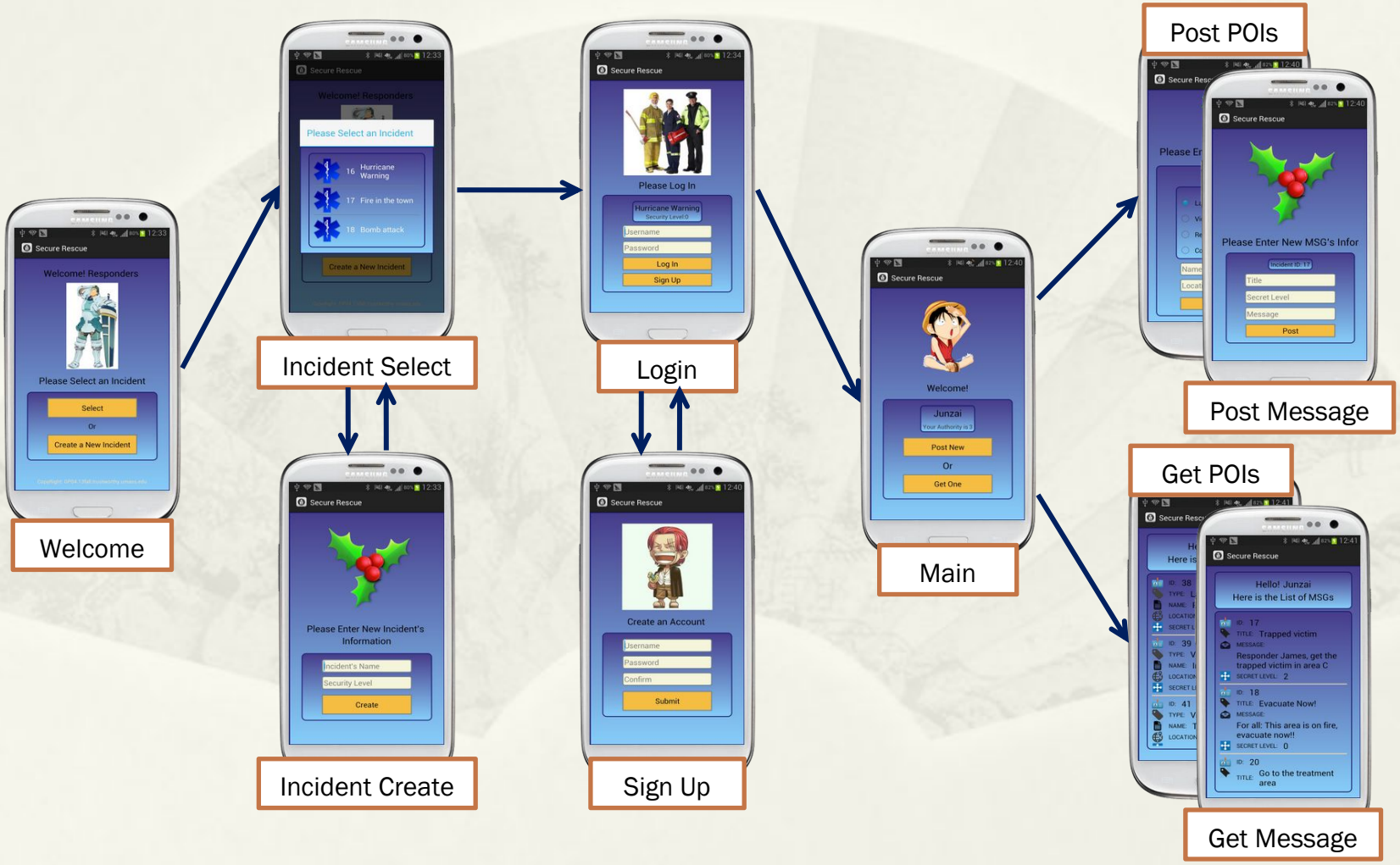
A compromised strategy to balance between responding time (server resources) and security provided.

Level	Access Control	Transmit Encryption	Database Protection
Low			t
Mediun			t
High	• Username • Password • Authentication • Authority	• Encrypted by a session key • Authenticator and ticket required	• Encrypted • Key never stored with data • Certification

Tips:

- Each incident has a security level separated from others
- The security level of an incident is decided by user when created
- Once determined, the security level cannot be changed

Client Application



Security Level Low

User only has to input a username

Password input field is invalidated

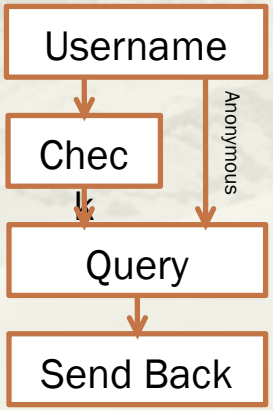
Request is sent to data server in plaintext

Authority is checked in Server

I'm Yiming, I want POI List

You're authority is 2, I can give you these POIs...

Data Server



Security Level Low

Anonymous

- Anonymous login is an option to the server admin
- Once it's turned on, user can input **ANY** username to receive data and is treated as authority 0 in server
- Only authenticated users can send data

Summary

- The simplest login procedure for users
- Do not always need to create an account
- Forgery is easy when you get someone's username
- Eavesdropping is also easy

Best used at: Alert broadcast, EMT training, etc.

Result

Security Level Medium

Security Level Medium

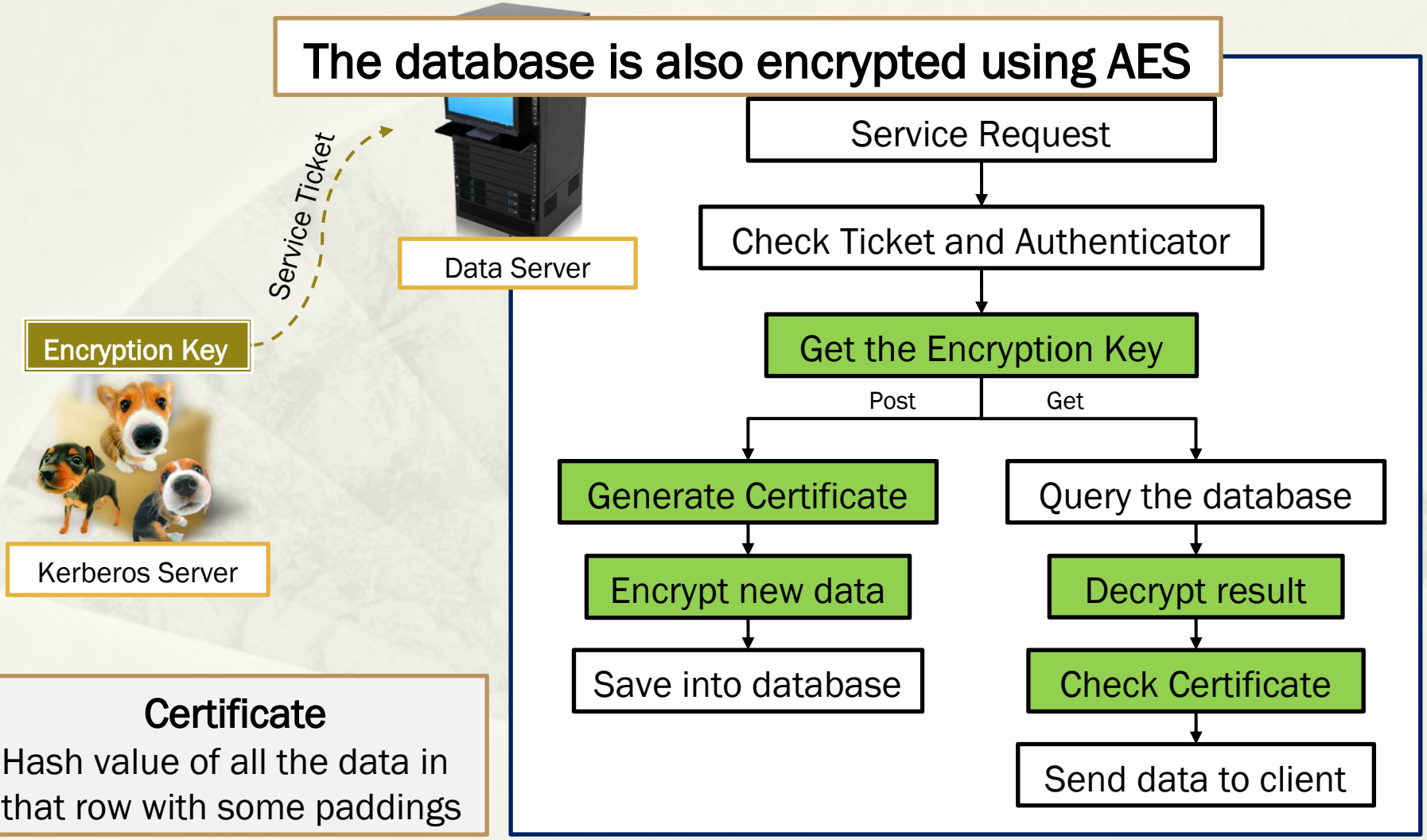
Result

Summary

- Kerberos Server and Data Server are separated
- Both user and server are authenticated through Kerberos procedure
- All communication are encrypted by a session key
- User's authority is passed from AS to Data Server through ticket
- In both GET and SEND processes, server will check the ticket and authenticator
- Data is stored in plaintext in the database

Best used at: Natural disaster, casual incidents, etc.

Security Level High



Certificate
Hash value of all the data in that row with some paddings

Security Level High

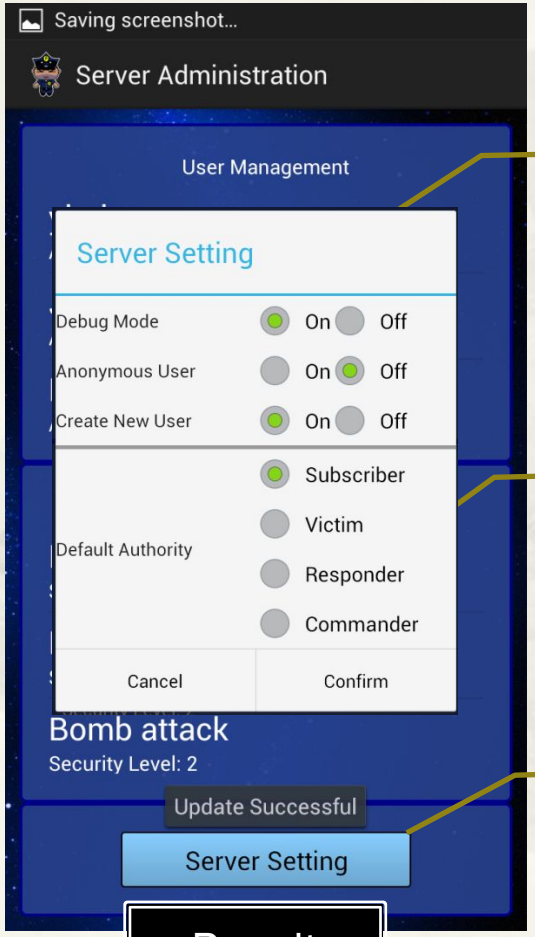
Summary

- Database is encrypted using AES
- Encryption key is never stored in the data server
- The data server is able to decrypt the data only when an authenticated user brings a ticket
- Certificate code helps detect the unauthorized changes of data
- An attacker cannot understand or forge anything even if he has the full access to the Data Server
- Most security resources can be put on Kerberos Server rather than Data Server

Best used at: Terrorist attack, aliens attack, etc.

Result

Administrator App



Result

Allows the administrator to manage the server

User Management

- View all the users and their authorities
- Assign them different authorities
- Delete a user (to be implemented)

Incident Management

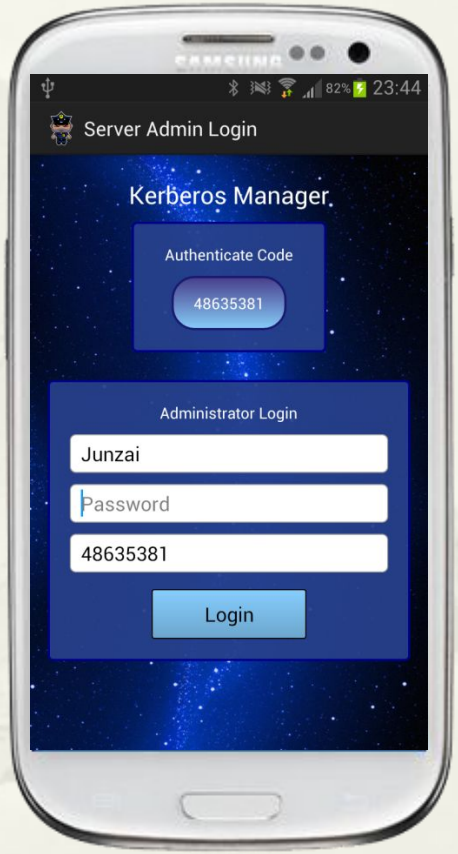
- View all the incidents
- Lock the incident to make it read-only
- Unlock the incident to make it editable

Server Settings

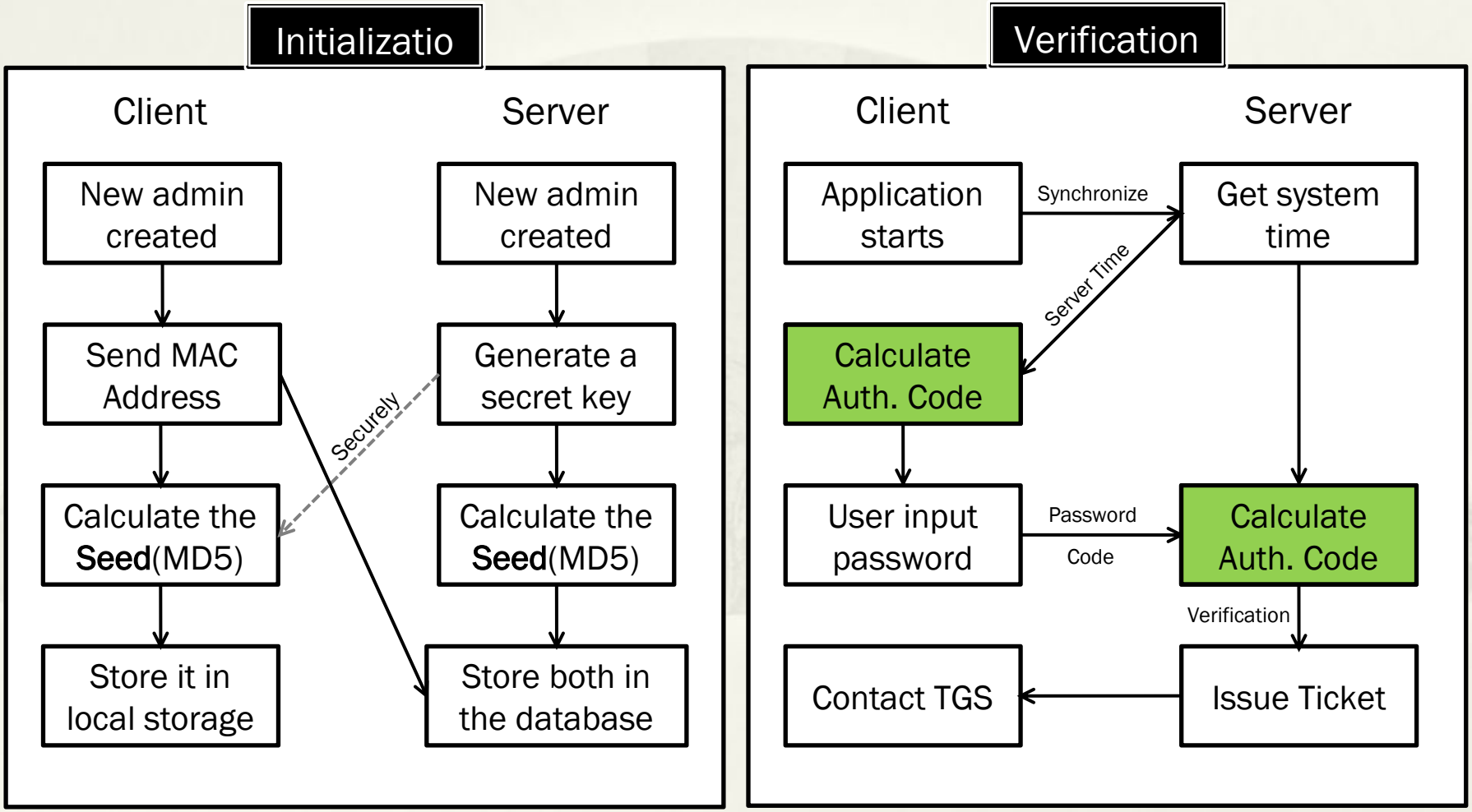
Administrator Login

Authenticate Code (Security Token)

- A 6 or 8 digits number code that is generated and refreshed by the client every 30 seconds
- The code is **unique** to every device and every user
- Server generates the same code separately and verifies the client's code **before** checking the password
- Administrator should use his own device in addition to having the password

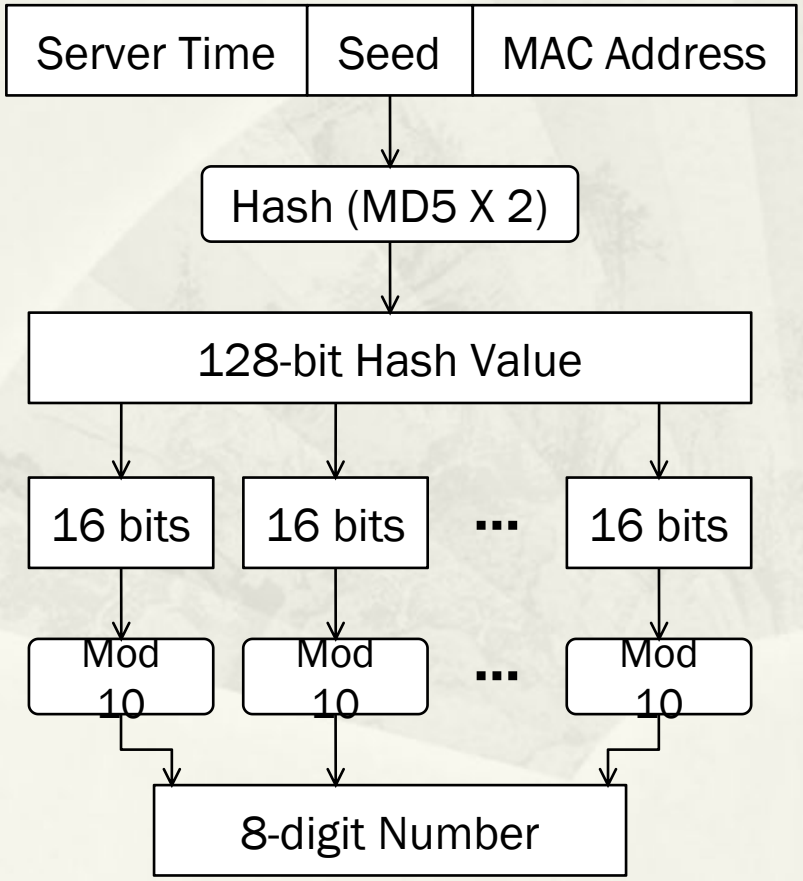


Authenticate Code



Authenticate Code

Code Calculation



Summary

- Seed
  - Only generated once when the app is used at the first time
  - Stored secretly in local and is not accessible for the user
- Authenticate Code
  - Is calculated separately and is never transmitted through network
  - Every code can only be used once
  - Is changed every 30 seconds to prevent replay attack
- MAC Address ensures one Seed is fixed to only one device
- An attacker cannot log into the server unless he/she gets both the device and password

Let's Attack

Sample Incidents

- Three Incidents with the same content but different security level from low to high
- Each incident contains: 2 landmarks, 4 victims, 1 responder and 1 commander

Sample Users

- A user named Apple with authority Victim
- A user named Android with authority Commander

Sample Attacks

- Username masquerading
- Network sniffing
- Data Server intrusion
- Brute-Force

Server Setting

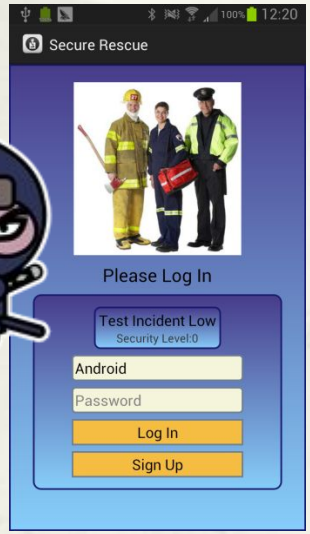
Anonymous: Allowed

Target Data

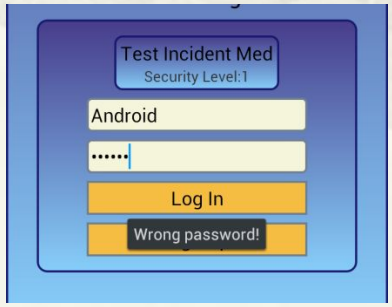
Name	Location
Landmark 1	Area 1
Landmark 2	Area 2
Victim 1	Area 5
Victim 2	Area 5
Victim 3	Area 7
Victim 4	Area 10
Responder 1	Area 5
Commander	Area 15

Username Masquerading

Security Level Low



Security Level Medium and High



Password is able to protect user's account from masquerading and offers authentication to server

Network Sniffing

Wireshark

- A software that allows user to see all traffic visible on the network
- Can be installed in the server, the router or other workstations in the same LAN (IP Spoofing)
- In the test, we login as the user Android and try to get all POIs

Network Sniffing

Security Level Low

Capturing from 无线网络连接 [Wireshark 1.10.3 (SVN Rev 53022 from /trunk-1.10)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: http && (tcp.port == 8080 || udp.port == 8080) Expression... Clear Apply Save Filter

No.	Time	Source	Destination	Protocol	Length	Info
805	131.976772	192.168.15.161	192.168.15.125	HTTP	211	GET /api/Auth?incidentTop=0 HTTP/1.1
806	132.001239	192.168.15.125	192.168.15.161	HTTP	731	HTTP/1.1 200 OK (application/json)
2939	177.229214	192.168.15.161	192.168.15.125	HTTP	412	POST /api/Service1?incidentId=19 HTTP/1.1 (application/json)
2940	177.240211	192.168.15.125	192.168.15.161	HTTP	1192	HTTP/1.1 200 OK (application/json)

```
⊞ Frame 2940: 1192 bytes on wire (9536 bits), 1192 bytes captured (9536 bits) on interface 0
⊞ Ethernet II, Src: IntelCor_73:a0:70 (9c:4e:36:73:a0:70), Dst: SamsungE_ee:48:46 (5c:0a:5b:ee:48:46)
⊞ Internet Protocol Version 4, Src: 192.168.15.125 (192.168.15.125), Dst: 192.168.15.161 (192.168.15.161)
⊞ Transmission Control Protocol, Src Port: http-alt (8080), Dst Port: 34407 (34407), Seq: 1, Ack: 347, Len: 1126
⊞ Hypertext Transfer Protocol
⊟ JavaScript Object Notation: application/json
  ⊟ Array
    ⊞ Object
    ⊞ Object
    ⊞ Object
    ⊟ Object
      ⊟ Member Key: "str_id"
          String value: 60
      ⊟ Member Key: "str_POIType"
          String value: 1
      ⊟ Member Key: "POIName"
          String value: Victim 2
      ⊟ Member Key: "POILocation"
          String value: Area 5
      ⊟ Member Key: "str_SecretLevel"
          String value: 1
      ⊞ Member Key: "isEncrypted"
    ⊞ Object
    ⊞ Object
    ⊞ Object
    ⊟ Object
      ⊟ Member Key: "str_id"
          String value: 65
      ⊟ Member Key: "str_POIType"
          String value: 3
      ⊟ Member Key: "POIName"
          String value: Commander
      ⊟ Member Key: "POILocation"
          String value: Area 15
      ⊟ Member Key: "str_SecretLevel"
          String value: 3
      ⊟ Member Key: "isEncrypted"
          False value
```

```
00d0  34 34 3a 34 39 20 47 4d  54 0d 0a 0d 0a 5b 7b 22   44:49 GM T....[{"
00e0  73 74 72 5f 69 64 22 3a  22 35 37 22 2c 22 73 74   str_id": "57","st
00f0  72 5f 50 4f 49 54 79 70  65 22 3a 22 30 22 2c 22   r_POITyp e":"0","
0100  50 4f 49 4e 61 6d 65 22  3a 22 4c 61 6e 64 6d 61   POIName" :"Landma
0110  72 6b 20 31 22 2c 22 50  4f 49 4c 6f 63 61 74 69   rk 1","P OILocati
```

JavaScript Object Notation (json), 971 by... Packets: 7322 · Displayed: 4 (0.1%)

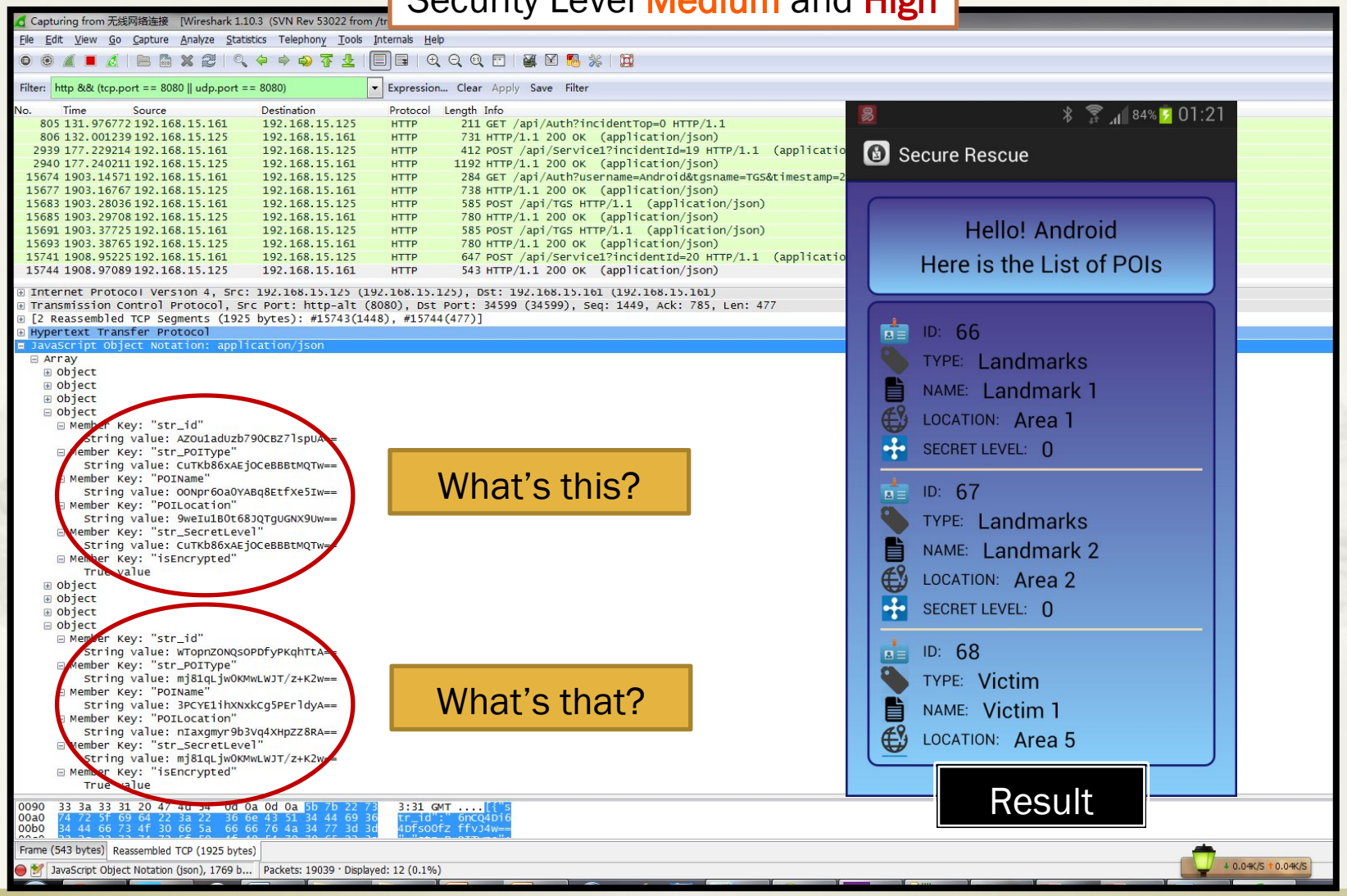
Victims

Commander

All information is transmitted in **plaintext**

Network Sniffing

Security Level Medium and High



Data Server Intrusion

```sql
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [Id]
      ,[POIType]
      ,[POIName]
      ,[POILocation]
      ,[SecretLevel]
      ,[IncidentId]
      ,[Certificate]
  FROM [DB_DataServer].[dbo].[TB_Service1]
  WHERE IncidentId = 19
```

Security Level Low and Medium

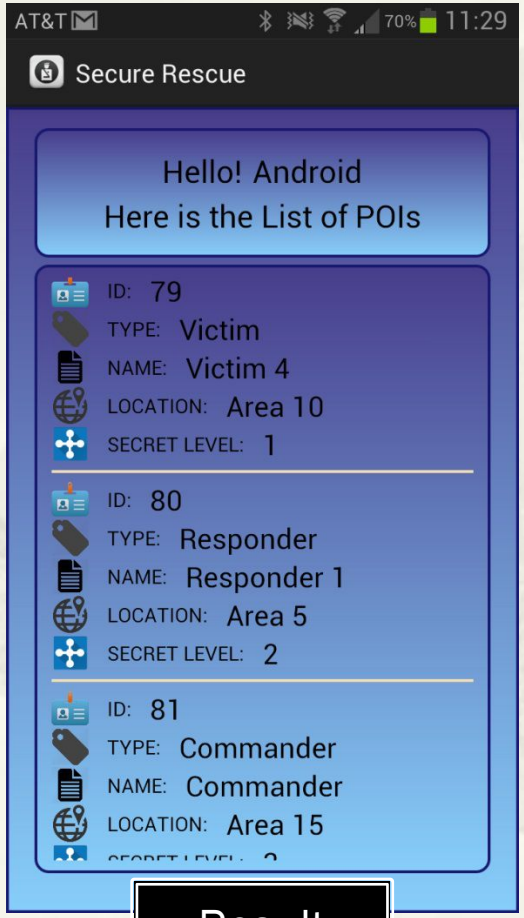
	Id	POIType	POIName	POILocation	SecretLevel	IncidentId	Certificate
1	57	0	Landmark 1	Area 1			
2	58	0	Landmark 2	Area 2			
3	59	1	Victim 1	Area 5			
4	60	1	Victim 2	Area 5			
5	61	1	Victim 3	Area 7			
6	62	1	Victim 4	Area 10			
7	64	2	Responder 1	Area 5			
8	65	3	Commander	Area 15			

	Id	POIType	POIName	POILocation	SecretLevel	IncidentId	Certificate
1	66	0	Landmark 1	Area 1	0	20	NULL
2	67	0	Landmark 2	Area 2	0	20	NULL
3	68	1	Victim 1	Area 5	1	20	NULL
4	69	1	Victim 2	Area 5	1	20	NULL
5	70	1	Victim 3	Area 7	1	20	NULL
6	71	1	Victim 4	Area 10	1	20	NULL
7	72	2	Responder 1	Area 5	2	20	NULL
8	73	3	Commander	Area 15	3	20	NULL

Security Level High

	Id	POIType	POIName	POILocation	SecretLevel	IncidentId	Certificate
1	74	0	YcOKcHPhiZ+4tC74MU/URA==	+EbEzktfLATzJHHwYwZaAg==	0	21	FCFA85A58C04EFEAE39B403DC19F3E09
2	75	0	bI6z5mwO/nAvoRv7rSGGxQ==	oWuohmp9Z+7oQ9tNq2+ZOA==	0	21	B5029395EAD9B1603CC597C0AA0855F1
3	76	1	uC4tpouh3NrdkKRwaBNVog==	NazOnTyVoozINTT6S7OOkw==	1	21	0D9441ED4DE022220F85ED3FD1D7B4F3
4	77	1	1LGp5VOnuWFqzHxFLRYrNA==	NazOnTyVoozINTT6S7OOkw==	1	21	71336341544CABCA01AA7685CAA812F4
5	79	1	I2oZIMAuRX3BLYJcj5rXLw==	gzvW87ekYDSCRrWk7ZvCHw==	1	21	D296DBF7039EAE71B4E80502273CFB8D
6	80	2	8DDnrd9x7KT1koU6bJSxvQ==	NazOnTyVoozINTT6S7OOkw==	2	21	73BF8B406759C14E1708AE333C210240
7	81	3	ZT1+fPwUVAI3ax4jteL4AA==	dTEYh4qrLcWPcPj2GtkjJQ==	3	21	2C8BD162169E95CEAD2E4CCACC255364
8	82	1	YmXynBhp6jh2c61GPPsMdQ==	+I+YoJvJwfsdrt3ZfVZalA==	1	21	42CBCBAE9A399E906A09A0BB79561159

Data Server Intrusion



Result

Summary

- The Data Server is encrypted using AES
- The key is stored in Kerberos Server
- The data is firstly decrypted using the database encryption key and encrypted using Kerberos session key
- Certificate helps to detect unauthorized change

Error Log

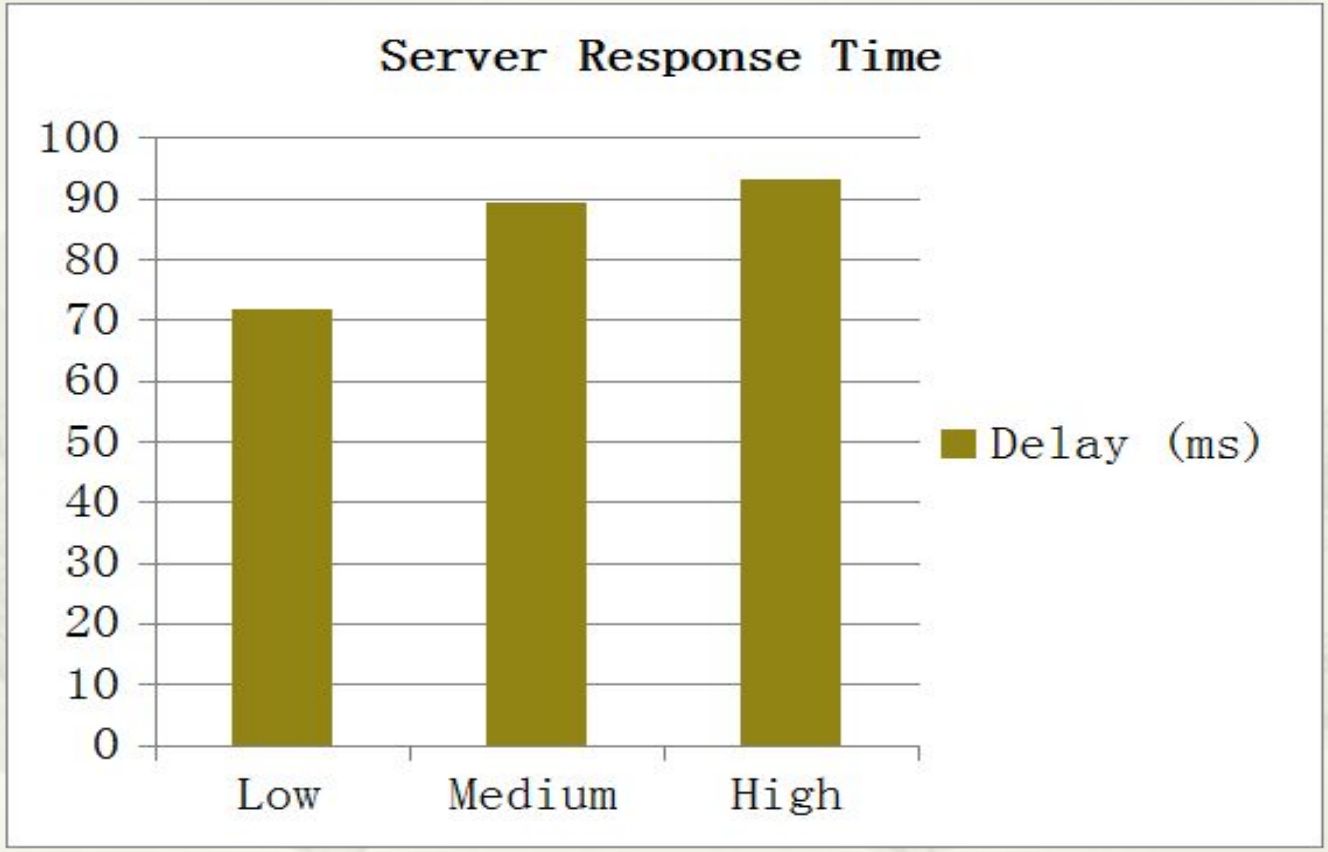
34	38	getPOIForLevel_2	0	Certificate failed

Brute-Force

Brute-Force?

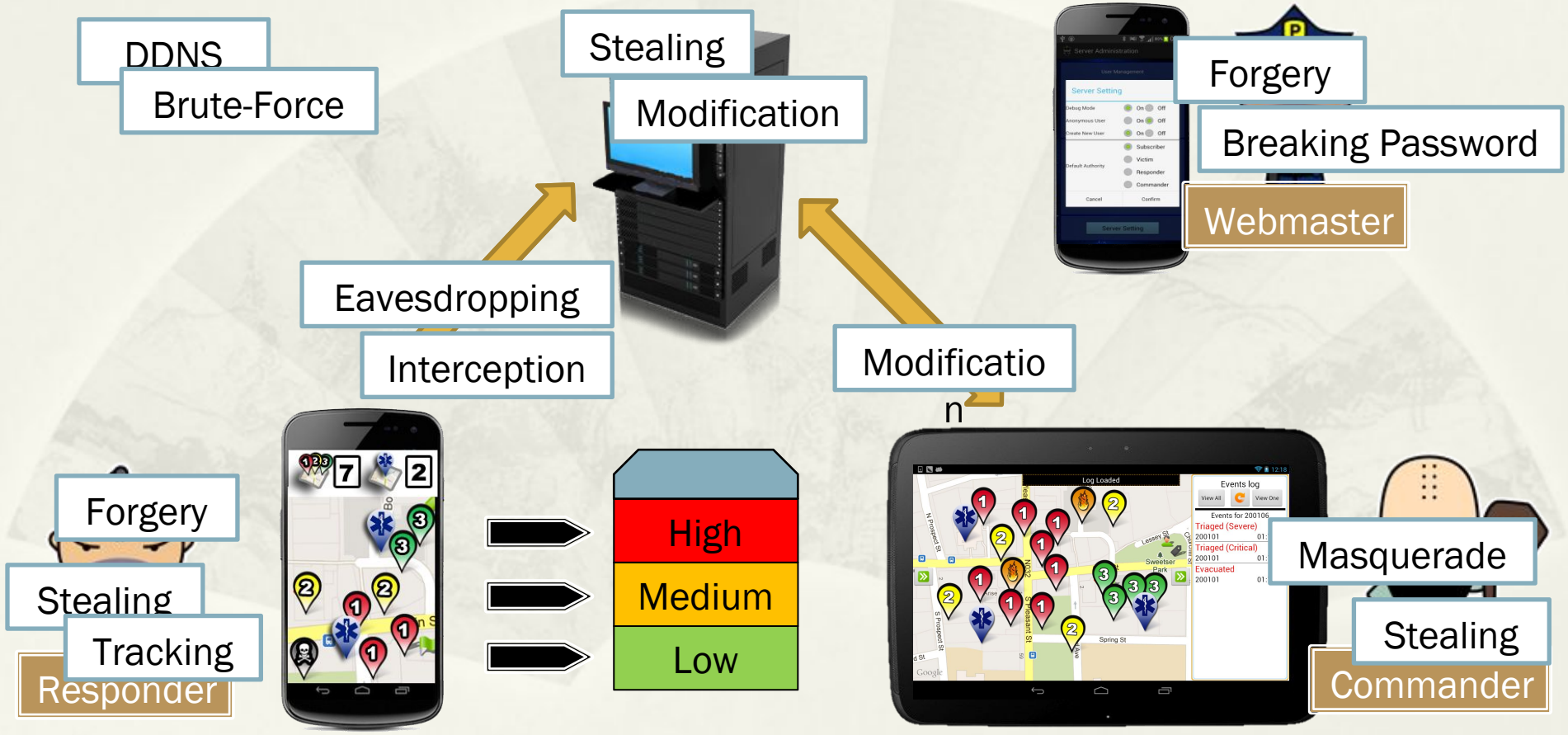
We have AES-128!

Performance



(Average time of 10 queries)

Protection We Provided



Future Work

- Ways to enhance Kerberos Server
- Extension of Authenticate Code
- System efficiency

Thank you!