Preventing Session Hijacking using Encrypted One-Time-Cookies

Renascence Tarafder Prapty, Shuhana Azmin, Md. Shohrab Hossain
Dept of CSE, Bangladesh University of Engineering and Technology
& Husnu S. Narman

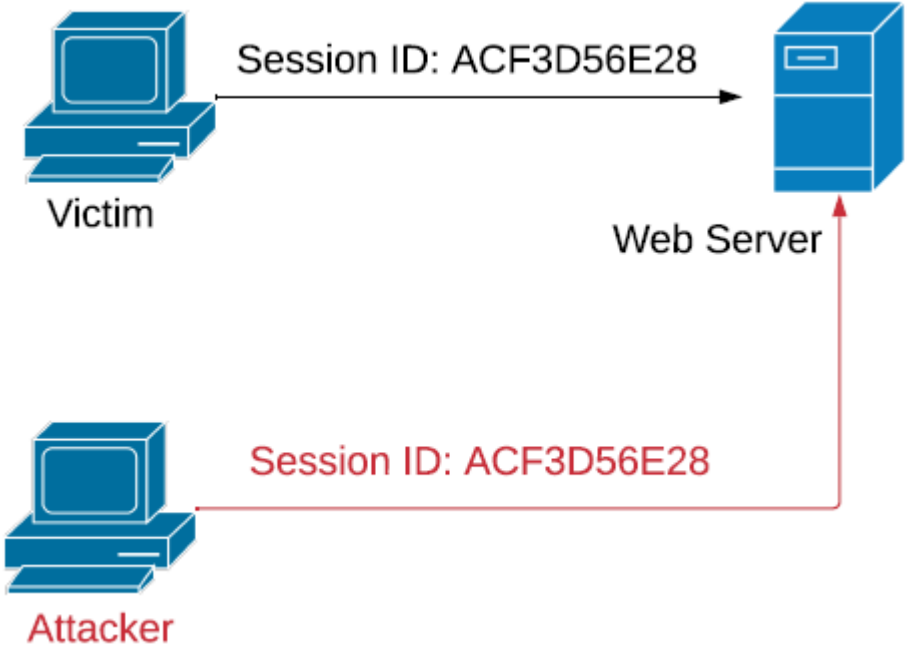
Presentation @ WTS 2020

Overview

- Session Hijacking and risks
- Existing works
- Proposed Architecture
  - Reverse Proxy Server
  - Cryptography Operations Module
- Details of Cryptography Operations
- Result
  - Security Analysis
  - Timing Analysis
- Summary

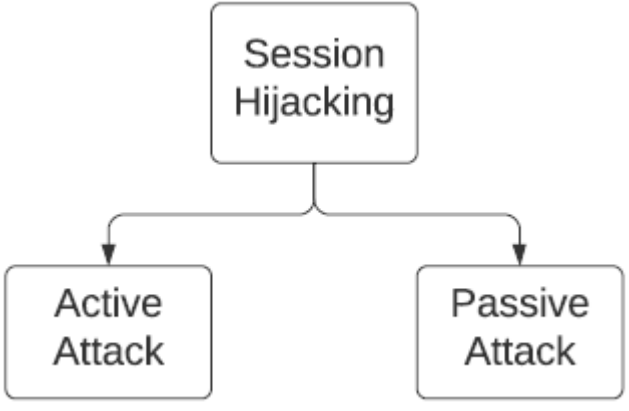
What is Session Hijacking?

- It is basically hijacking of sessions by intercepting the communication between hosts.
- The attacker usually intercepts the communication to obtain the roles of authenticated user or to gain access to information or services.

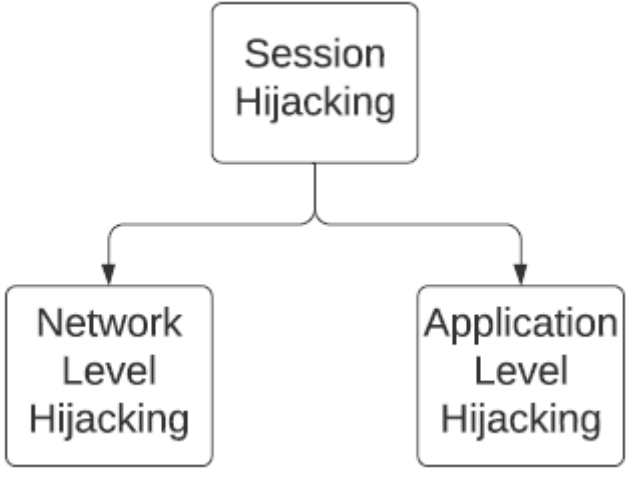


Different Types of Session Hijacking

- Based on activity of attacker



- Based on target level



What are the risks?

- Identity theft
- Information theft
- Loss of sensitive information
- Unauthorized modification of application

Existing Works

Can be classified into two groups

- Use of One Time Cookies(OTC)
  - OTC-based systems generate cookie per user request. It can prevent session replay attack but cannot ensure cookie confidentiality.
- Encryption of sensitive data in cookie
  - Encryption based systems can ensure cookie confidentiality but cannot prevent from session replay attack.

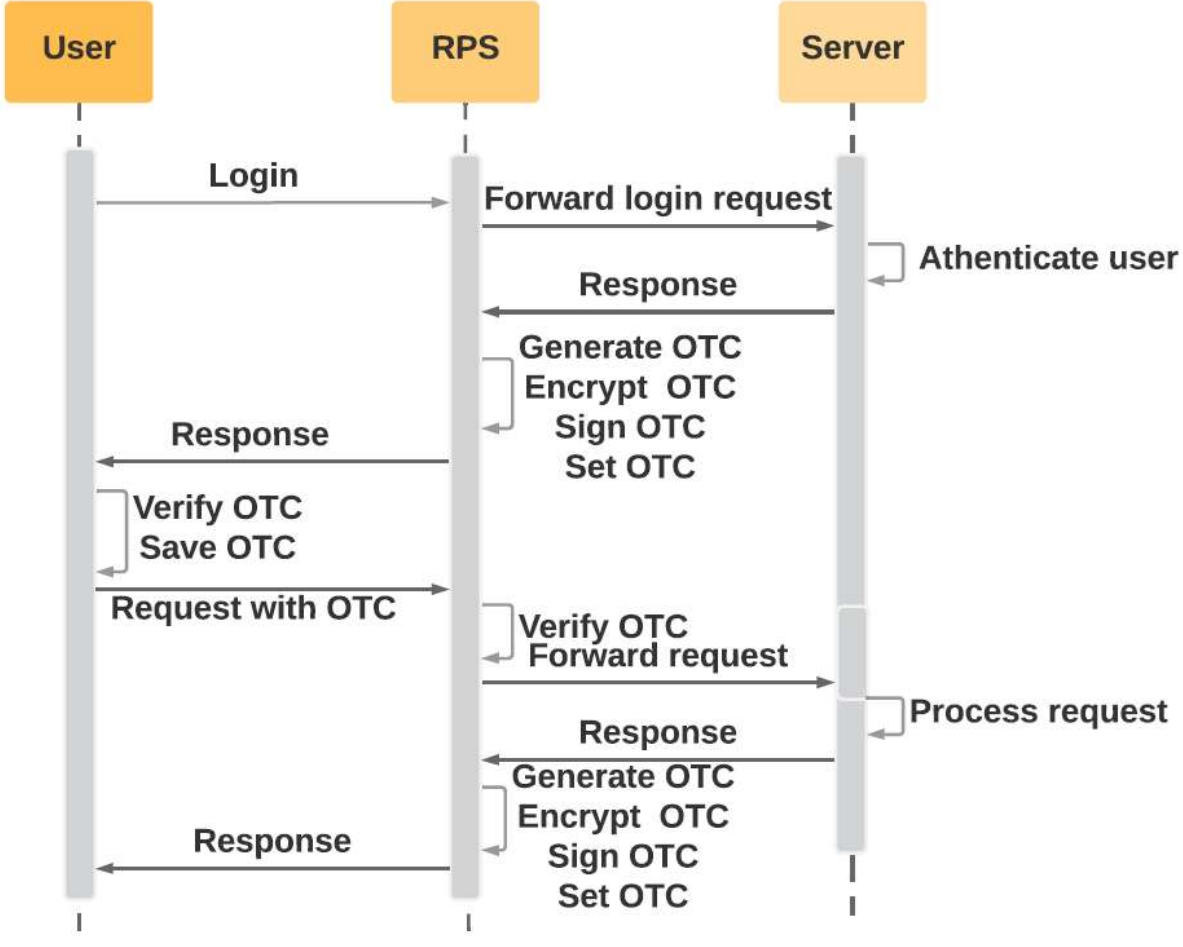
Proposed Architecture

- Consists of two modules:
  - **Reverse Proxy Server(RPS):** Issues and verifies One Time Cookies(OTC). The design of the reverse proxy server proposed in [1] is followed here.
  - **Cryptography Operations Module(COM):** Generates keys, encrypts and decrypts data, generates and verifies digital signature. Cryptography operations performed on session cookies in [2] provide a general guideline for the proposed module.

[1]A. M. Sathiyaseelan, V. Joseph, and A. Srinivasaraghavan, “A proposed system for preventing session hijacking with modified one-time cookies,” in International Conference on Big Data Analytics and Computational Intelligence. Chirala, India: IEEE, 23-25 March 2017, pp. 451–454.

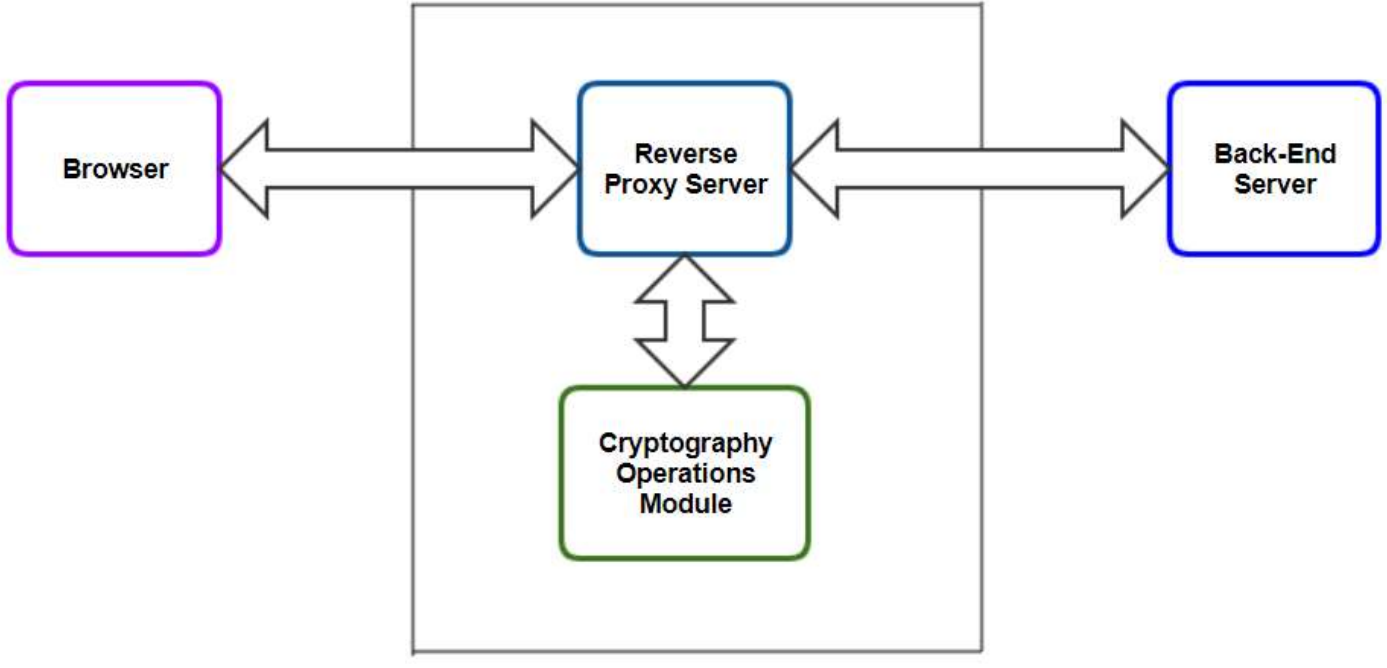
[2]W.-B. Lee, H.-B. Chen, S.-S. Chang, and T.-H. Chen, “Secure and efficient protection for HTTP cookies with self-verification,” International Journal of Communication Systems, vol. 32, no. 2, 2019.

Proposed Architecture



Roles of Reverse Proxy Server

- Collection of IP address and browser fingerprint from the client side
- Generation of a session ID and OTC
- Matching IP address, browser fingerprint and session ID along with OTC



Roles of Cryptography Operations Module

- Generating long term asymmetric key pair
- During OTC Issue Phase:
  - Breaking OTC into non-sensitive component (C_{i1}) and sensitive component (C_{i2})
  - Selection of different secret parameter (k) for each OTC
  - Generation of Symmetric Key (SK) from C_{i1} and k
  - Encryption of C_{i2} using SK
  - Generation of digital signature for this partially encrypted OTC

Roles of Cryptography Operations Module

- Verification of digital signature by browser
- During OTC Verification Phase:
  - Retrieving secret parameter (k) from the digital signature during verification of OTC
  - Reconstruction of Symmetric Key (SK) from k and non-sensitive information(C_{i1}) during verification of OTC
  - Detection of any modification in the OTC sent from the client

Details of Asymmetric Key Pair Generation

RSA algorithm has been implemented to generate Asymmetric Key Pair. The implementation process is described below:

- Randomly selecting a large Prime Number p
- Calculating a Primitive Number $g \in GF(p)$
- Randomly selecting Private Key $x \in [1; p-1]$
- Calculating Public Key $y = g^x \bmod p$

Details of Symmetric Key Generation and Encryption

- During generation of each OTC, a secret parameter k is calculated such that it fulfills following conditions:
  - $k \in [1; p - 1]$
  - $gcd(k; p - 1) == 1$.
- Non-sensitive content of OTC and k are concatenated and hashed using the SHA256 algorithm to generate a symmetric key. This process can be expressed as follows:
  - $SK = h(C_{i1}||k)$
- Symmetric key is used to encrypt sensitive content. It can be described as $T_i = E_{SK}(C_{i2})$. Here $E_{SK}()$ is the Encryption function.

Details of Digital Signature Creation and Verification

- Digital signature (r,s) of OTC is created using the following equations:
  - $r = g^k \bmod p$
  - $s = x * (r + h(C_{i1}||Ti) - k \bmod (p - 1)$
- C_{i1}; t_i; r; s are sent to client as part of the OTC. To check the authenticity of OTC, the client's browser can verify the digital signature using the following equation:
  - $y^{r+h(Ci1||Ti)} = r * g^s \bmod p$

Details of Symmetric Key Reconstruction and Decryption

- Client's browser includes provided OTC in next request.
- During verification of an OTC, k is retrieved from digital signature using following equation:
  - $k = x * (r + h(C_{i1}||T_i) - s \bmod (p - 1)$
- Symmetric key is reconstructed using following equation:
  - $SK = h(C_{i1}||k)$
- Symmetric key is used to decrypt the encrypted sensitive content. It can be described as $C_{i2} = D_{SK}(T_i)$. Here $D_{SK}()$ is the Decryption function.

Results: Security Analysis

- **Ensuring confidentiality:**
  - The sensitive part of the OTC is encrypted by secret key SK.
  - SK is not stored in RPS or transmitted to the Client over the network.
  - Hence, any eavesdropper cannot sniff it from the transmission link and confidentiality is maintained.

Results: Security Analysis

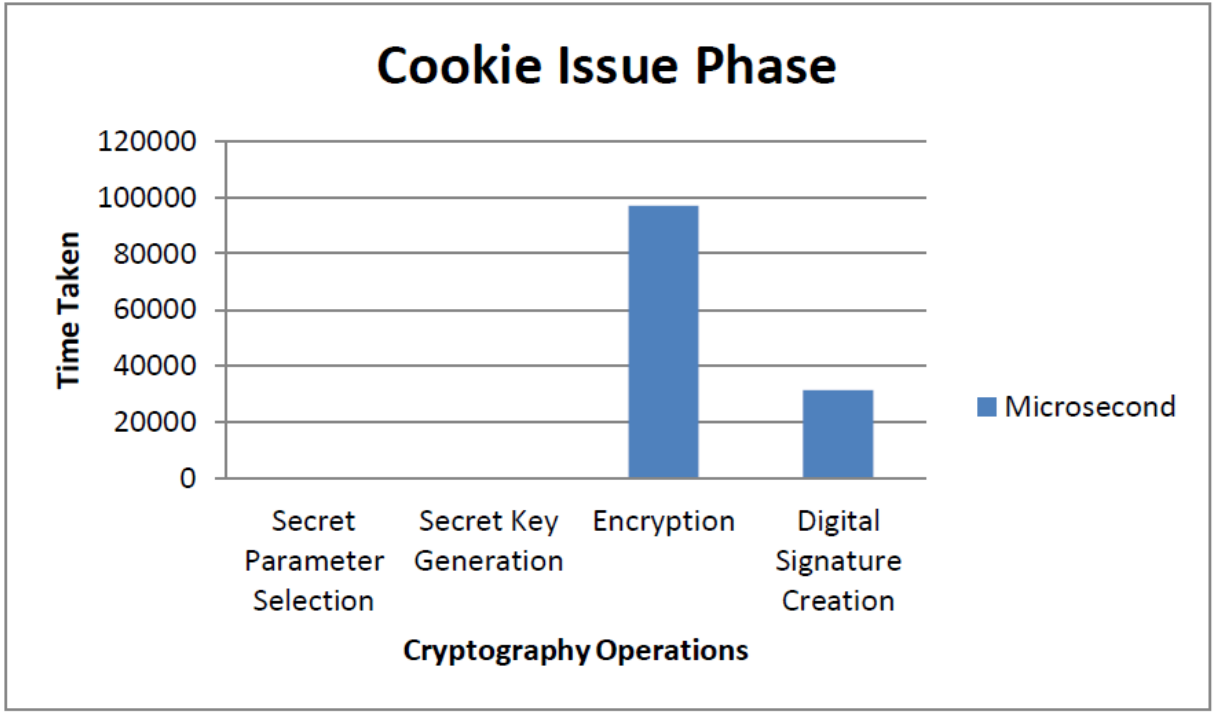
- **Ensuring authenticity:**
  - RPS signs OTC with its private key.
  - Client’s browser can use the public key of RPS to check the authenticity of OTC.
  - If attacker forges a signature without using the private key of RPS, the signature verification fails.
- **Ensuring integrity:**
  - Change in any part of OTC results in wrong Secret Key reconstruction.
  - As a result decryption operation fails and change is detected.

Results: Security Analysis

- **Prevention against replay attack:**
  - For each request, an OTC is generated by RPS.
  - RPS matches session ID and expiry time of OTC returned from browser with expected values.
  - Hence an attacker cannot perform replay attack by using an expired or already used OTC with a new request.

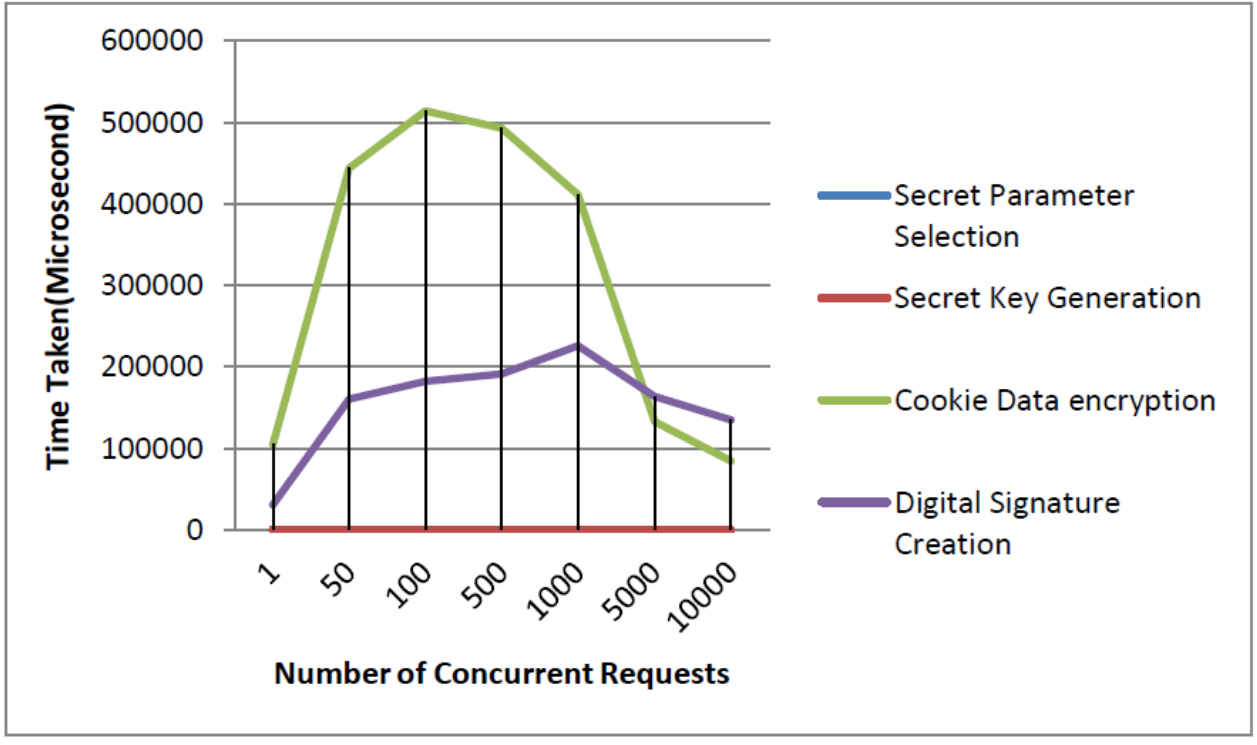
Results: Timing Analysis

- **During OTC Issue Phase:** Breakdown of time required for different operations



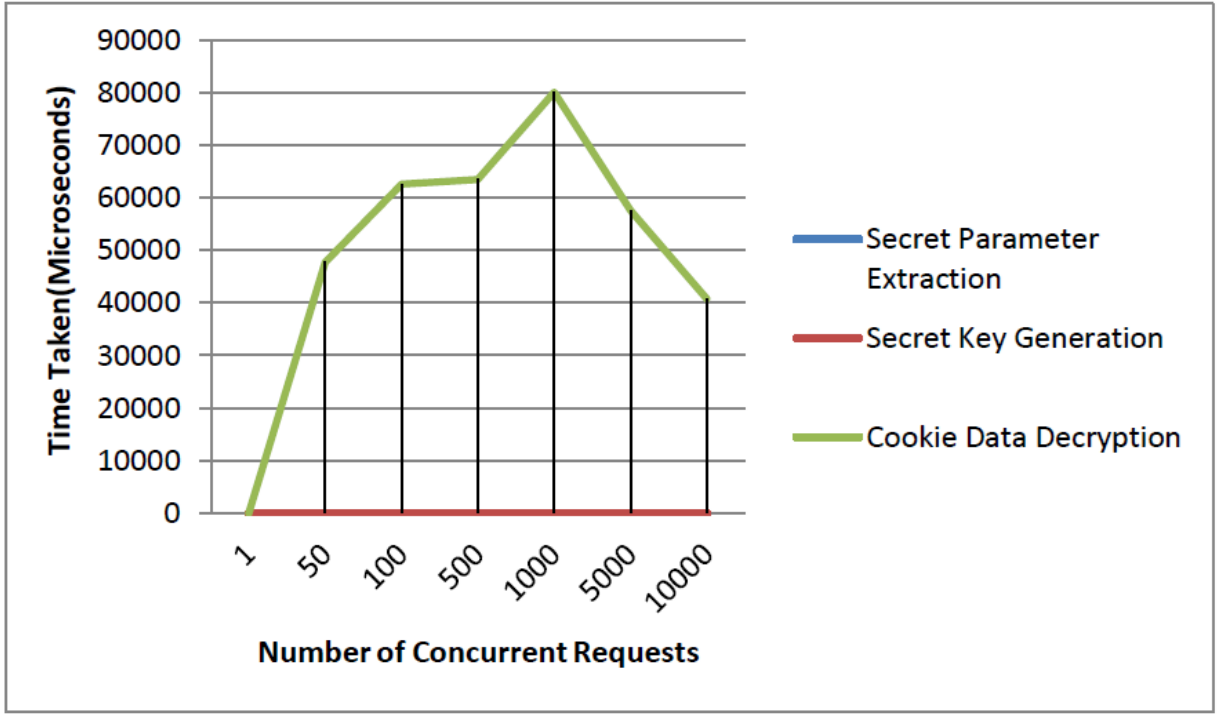
Results: Timing Analysis

- **During OTC Issue Phase:** Time required for different numbers of simultaneous requests



Results: Timing Analysis

- **During OTC Verification Phase:** Time required for different numbers of simultaneous requests



Summary

- Encrypted one time cookies to prevent session hijacking
- One Time Cookies issued and verified by Reverse Proxy Server
- Encryption and decryption of Sensitive information
- Generation and verification of digital signature
- Security analysis to ensure confidentiality, authenticity, integrity and to prevent replay attack
- Timing analysis of OTC Issue Phase and OTC Verification Phase

Thank you!
Any questions?