Manish Mehta
Security Engineer
Jan 11, 2018 @ RWC 2018
Disclaimer

- Design discussions and statements in this presentation do not necessarily reflect Netflix’s future business plans
- Parts of this presentation are under a US patent (pending)
AWS console breach leads to demise of service with “proven” backup plan

Code Spaces closes shop after attackers destroy Amazon-hosted customer data.

Dan Goodin - 6/18/2014, 2:12 PM

The hackers reportedly gained access to a private Git repository, then accessed company servers, then contacted UC Berkeley computer science engineering students to see if they had any leads to the attackers. They should be able to track it down.
public DBResult getEmployeeData() {

    String host = "database.example.com";
    String username = "operator";
    String password = "myCrazyLongPasswordThatIsUnpredictable";
    String query = "SELECT * from employee;";

    DBConnection connection = new DBConnection(host, username, password);
    connection.execute(query);

    // Format the output and return

}
public DBResult getEmployeeData() {

    String host = "database.example.com";
    String username = "operator";
    String encPassword = "EBEABKihxG01UEe50JXpazdhUH5ijuL6a15VmIRBZi+eizn6+IXJTcKo7";
    String password = decrypt(encPassword);

    String query = "SELECT * FROM employees";

    DBConnection connection = new DBConnection(host, username, password);
    connection.execute(query);

    // Format the output and return
}
Let's build a story

```java
public DBResult getEmployeeData() {
    String host = "database.example.com";
    String username = "operator";
    String encPassword = "EBEABKihxGO1UEe50JXpazdhUH5ijuL6a15VmIRBZi+eizn6+IXJTCko7";
    String password = decrypt(encPassword);
    String query = "SELECT * FROM employee;"

    DBConnection connection = new DBConnection(host, username, password);
    connection.execute(query);

    // Format the output and return
}
```
Story at Netflix

git

Jenkins

Spinnaker

Developers

Key Server

Application

HSM?
Story at Netflix

Decryption Steps

1. Authenticate Requestor
2. Decrypt the Secret using the right key
Step 1: Authenticate Requestor

Requestor’s Identity

1. Users
   - mTLS or Oauth
   - Identity Bootstrapped thru User Identity Provider

2. Applications (AWS VMs/Containers)
   - mTLS
   - Identity Bootstrapped thru AWS Metadata service
Step 1: Authenticate Requestor

Identity Bootstrapping for Applications (AWS VMs)

- Use AWS Metadata Service as Root-of-Trust


MIAGCSqGSIb3DQEHAqCAMIACAQExDzANBglghkgBZQMEAgEFADCAKBgkqhkiG9w0BBwGgCSABIIBsnsKICAIZGV
2cGF5UHJvZHvdENvZGVzIiA6IG51bWgSSCgkIwQYDWRMRAwECAwIBAQIBAQPQgDCCwoDCCwHBgYq
Step 1: Authenticate Requestor

AWS Metadata Service Output

```json
{
  "data": {
    "devpayProductCodes": null,
    "privatelp": "100.66.43.244",
    "availabilityZone": "us-east-1e",
    "accountId": "179727202194",
    "version": "2010-08-31",
    "instanceId": "i-0f8392bc5987c08b1",
    "instanceType": "m3.2xlarge",
    "imageId": "ami-e60c95f1",
    "pendingTime": "2016-08-12T22:28:09Z",
    "architecture": "x86_64",
    "kernelId": null,
    "ramdiskId": null,
    "region": "us-east-1"
  },
  "signature": "DqktfKuv2r8j ...
  JqIYWS0aMoFjZhYMg4G"
}
```

AWS describeInstance Output

```json
{
  "architecture": "x86_64",
  "class": "com.amazonaws.services.ec2.model.Instance",
  "imageId": "ami-e60c95f1",
  "instanceId": "i-0f8392bc5987c08b1",
  "instanceType": "m3.2xlarge",
  "launchTime": 1471040889000,
  "privateDnsName": "ip-100-66-43-244.ec2.internal",
  "privateIpAddress": "100.66.43.244",
  "securityGroups": [],
  "tags": [
    {
      "aws:autoscaling:groupName": "infocrypt-v002",
    }
  ],
  "vpcId": "vpc-12345"
}
```

Details on this in
1. Enigma 2017 Conference
2. Future:NET 2017 Conference
Step 2: Decrypt

Requirement

Each Group of User(s) and Application(s) MUST have at least one unique key

For e.g.

\[ K_1 \text{ for } G_1 = [ \text{ Alice }, \text{ Bob }, \text{ Application}_1, \text{ Jenkins}_1 ] \]
\[ K_2 \text{ for } G_2 = [ \text{ Eve }, \text{ Application}_2, \text{ Application}_3 ] \]

...
Let’s talk scale

If we have $N$ Users and $M$ Applications, maximum # of groups is …

$$\sum_{k=1}^{M+N} \binom{M+N}{k} = 2^{(M+N)} - 1$$

For $N = 10$ and $M = 10$, the number is 1 Million+
For $N = 12$ and $M = 12$, the number is 16 Million+
But, why complicate?
Define our Goals

Goal

• Secret MUST NOT ever be readable in clear except for the creator and intended consumers (Not even the Decryption Service)

Stretch Goals

• Offline Encryption of Secrets SHOULD BE supported
• Decryption Service’s ability to observe usage pattern SHOULD BE limited

Constraints

• # of Keys should scale
• # of Request should scale
Goals - Visually

Offline

Secret Creator

Code

M → C

Online

Secret Creator

Secret Consumer

App

C M

Secret Decryptor

M

→

→
Our Solution - Inspiration

How to Date Blind Signatures

Masayuki Abe  Eiichiro Fujisaki
abe@isl.ntt.jp  fujisaki@sucaba.isl.ntt.jp

Tel: +81 468 59 2570  Fax: +81 468 59 3858
NTT Information and Communication Systems Laboratories
Nippon Telegraph and Telephone Corporation
1-2356 Take, Yokosuka-shi, Kanagawa, 238-03 Japan

Abstract. A blind signature provides perfect confidentiality to a message and signature pair. Due to this feature, the blind signature has one downside; the signer can not assure himself that the blinded message accurately contains the information he desires. In a practical sense, it is essential for the signer to include some term of validity in the signing message to prevent abusing. Of course the term must not violate the confidentiality of the message. This paper discusses partial blinding of a signed message. We consider RSA and it is proved that forging the proposed scheme by multiple signing is as difficult as breaking RSA. The strategy can be also applied to those blind signature schemes that use a trapdoor function. An electronic cash system is shown as an application of the proposed scheme. Unlike most privacy-protected electronic cash system, it successfully minimizes the growth of the bank's database.

Our Solution - Setup

Let $G_{ID}$ be group ID with length $(k - 2)$ bits.
Let $\tau(G_{ID}) = 2^{k-1} + 2G_{ID} + 1$
That is, $\tau(G_{ID_i})$ does not divide $\tau(G_{ID_j})$ where $i \neq j$

Choose two large primes $p$ and $q$ such that $s_i \nmid \lambda$ for all prime $s_i$ $(3 \leq s_i \leq 2^{k-1} - 1)$
Where $\lambda$ is the LCM of $p - 1$ and $q - 1$

Choose public prime exponent $e \geq 2^k - 1$
Compute $d$ such that $ed = 1 \mod \lambda$
Our Solution – In Action

Encrypt

\[ C = M^{e \cdot \tau(G_{ID})} \mod N \]

Blind

Choose blinding factor \( R < N \)

\[ Z = C \cdot R^{e \cdot \tau(G_{ID})} \mod N \]

Decrypt

Compute

\[ d_{G_{ID}} = \frac{1}{e \cdot \tau(G_{ID})} \mod \lambda \]

\[ \phi = Z^{d_{G_{ID}}} \mod N \]

\[ M = \frac{\phi}{R} \mod N \]

Recover

\[ M \]
Padding

- OAEP, KEM

- Since the Decryption step is after Authentication, it is not practical for attacker to use it as Decryption Oracle without getting noticed.
Our Solution vs. Goals

Goal

● Secret MUST NOT ever be readable in clear except for the creator and intended consumers (Not even the Decryption Service)

✓ Blind Decryption Service behind Authentication

Stretch Goals

● Offline Encryption of Secrets SHOULD BE supported
● Decryption Service’s ability to observe usage pattern is limited

✓ Asymmetric system provides offline Encryption and Blinding limits Decryption Service’s visibility

Constraint

● # of Keys should scale
● # of Request should scale
✓ Stateless system with only 1 private key - Scalable
Taking it a step further

• $G_{ID}$ is just a positive integer of $(k - 2)$ bits
• It does not have to look like
  $G_1 = [\text{Alice}, \text{Bob}, \text{Application}_1, \text{Jenkins}_1]$
• Instead, it can look something like
  $G_1 = <\text{signed policy document with ID}>$
Other Constructions

• Aware of


• Other suggestions are welcome!
Next Steps

Keep looking for better underlying scheme

- Better Provable Security Guarantees
- Multi-party Blind Decryption
- PQ-resistant scheme
Resources

• Enigma 2017 Talk on Bootstrapping Identities
  https://www.youtube.com/watch?v=15H5uCj1hlE

• Future:NET 2017 Talk on Application Identity
  https://www.youtube.com/watch?v=g2efknf-HXQ

  https://doi.org/10.1007/BFb0034851
Thank you.

*(we are hiring)*

mmehata@netflix.com