**COMP9337 2022T1 Assignment DIMY**

W18B2 Haozhe Chen z5142012

W18B2 Hanqing Guan z5213081

1. **Introduction**

In this Assignment, we use a network of server nodes and multiple user nodes to track the diagnosis of each user and alert users who may be in close contact with diagnosed patient.

* 1. **Dimy client node**

Dimy.java is the client node which contains three threads inside of it.

The first thread is the Ephemeral ID thread which used to keep building new ephID every 15 sec and split each ephID into 5 shamir share. Each shamir share will be sent every 3 sec, and this message has 50% probability to be dropped.

图示

描述已自动生成

The second thread is the UDP receive thread. This thread contains three main functions, it will keep listening UDP package with UDP socket, and verify (reconstruct) the shamir secret once receiving three shamir shares from one client node. If the shamir secret has been reconstructed successfully, it will use Diffle-Hellman to exchange the ephID as the public key and get encounter ID as shared secret. After that, encID will be stored into a DBF bloom filter.

图示

描述已自动生成

The third thread is the DBF thread. This thread keep a bloom filter array which size is six. For every 90 sec, one of the array slot will be initiated as a new DBF. For every 9 min, the whole bloom filter array will be packed into a new bloom filter called QBF.

图示

描述已自动生成

* 1. **DIMY server node**

DimyServer.java is the server node which contains only one thread. This thread is used to keep receiving TCP socket message.

Dimy server will receive two types of bloom filter, QBF and CBF. If it received CBF, it means someone is diagnosed as a positive patient, server node will store this CBF. If it received QBF, it will try to check whether this QBF has intersection with certain CBF stored.

1. **Features**

|  |  |  |
| --- | --- | --- |
| **Task** | **Feature** | **Implementation** |
| 1 | EphID generation | **EphID.EphemeralID.generator()** |
| EphID is generated by Elliptic Curve. EC will output a keypair, public key will be used as EphID, private key will be kept by client node which will be used in Diffle-Hellman. |
| 2 | K-out-of-n Shamir | **Shamir package** |
| Shamir package code is borrowed from network. It can accept a BigInteger class secret, split and recover the Shamir share. |
| 3 | Shamir shares broadcast | **EphID.EphemeralID.run()**  **EphID.EphemeralID.broadcastShares()** |
| The split result of Shamir secret will be broadcast to everyone. |
| 4 | Reconstruct EphID | **UDP.UDPReceive.verify()** |
| Node will use send received Shamir share into verify function to get recovered Shamir secret. |
| 5 | Diffle-Hellman  Create EncID | **UDP.UDPReceive.DH()**  **UDP.UDPReceive.run()** |
| Once EphID has been reconstructed, client node will exchange their EphID (public key), and use other’s public key and own private key to generate the shared secret with ECDH algorithm. |
| 6 | Encode EncID into DBF | **BF.DBF.insert()** |
| The Bloom Filter Library is Google.guava. We use public interface **Bloomfilter.put** to add EncID into an existed bloom filter. |
| 7 | Create DBF | **BF.DBF.run()**  **BF.DBF.newDBF()** |
| Every client node maintains a DBFs array which size = 6. DBF created every 90s. |
| 8 | Create QBF | **BF.DBF.run()**  **BF.DBF.newQBF()** |
| QBF will be generated every 9min from current DBFs array. |
| 9 | QBF check | **TCP.TCPObjSend.sendQBF()**  **Helper.Helper.intersectCheck()** |
| Once server node receive QBF, it will try to match the local stored CBF, to check whether this client node close encountered diagnosed patient. |
| 10 | Upload CBF | **TCP.TCPObjSend.sendCBF()** |
| Sever will store CBF locally with Guava public interface **putAll.**  Server can distinguish CBF or QBF by TCP packet message send together with bloom filter. |

1. **Discussion**
   1. **ECDH**

We use Elliptic Curve to generate key pair, public key is EphID, private key is used when Diffle-Hellman key exchange.

* 1. **Shamir share broadcast**

Shamir share broadcast format: **|**shamir**|**H(EphID)**|**Prime**|**Shamir share**|**

*Shamir:* packet header, used to distinguish packet type (Shamir or DH)

*H(EphID):*Hash of EphID, node can use this to check whether reconstruction is successful. Also, this can be used to distinguish whether this broadcast message is sent by certain node itself.

*Prime:*Used in Shamir reconstruct.

*Shamir share:*Client node will input these shares to Shamir package to reconstruct EphID.

Once client node received three Shamir share packets from same node (This can be distinguish by check **H(EphID)** field of Shamir package). Then client node will input these Shamir shares into **Shamir.combine** to get the reconstruct result. Node can compare the hash of result with **H(EphID)** field in UDP packet.

* 1. **DBFs**

As mentioned in task7 and 8, each client node will maintain a bloom filter DBFs which size = 6 by Java thread. The array will update one of its slot every 90s. So for every 6min (90s \* 6), the entire array will be completely updates and QBF can be created by this array. Thread will also maintain an index counter to record which bloom filter of array is currently using.

* 1. **Possible improvement**

The TCP communication can be encrypted by some algorithm, for example: PGP. For example, when client node sends CBF or QBF to server node, client node will firstly send a TCP message “CBF” or “QBF” to indicate whether this bloom filter is CBF or QBF.

When improve, client node may attach a nonce to this message and use PGP to encrypt it. So that the integrity can be achieved.

* 1. **Borrowed Code**

1. The Shamir package is borrowed from network with a little change. Dimy will mainly use split and combine function of this package.
2. Bloom Filter is Google Guava bloom filter library.

*Notice: Since there is no public interface to set number of hashes of guava, Gunter said there is no need to meet this requirement.* <https://webcms3.cse.unsw.edu.au/COMP9337/22T1/forums/2815653>

1. **Attack**
2. For UDP between client nodes: Attacker can get Shamir share message, make a little temper and re-broadcast to same node, so that the EphID cannot be reconstruct, because node cannot distinguish whether this Shamir share message has been tempered before. With this attack, client node which should be marked as a “close contact” node will be not marked.
3. For TCP between client nodes and server node. Attacker may get the QBF sent out by diagnosed client node. However, since client node will firstly send out a TCP message (QBF or CBF) to indicate the Bloom filter type. Attacker can send “CBF” message but with a QBF followed. This attack will make the good node recognized as a diagnosed node (since only diagnosed can send CBF).