COMP4337 Assignment Spec notes

* **Main task – make digital contact tracing protocol DIMY (Did I Meet You)**
  + Using Rasp Pi4 & PC running Linux.
* Need to learn
  + Diffie-Hellman
  + Shamir Secret Sharing
  + Hashing
  + Bloom Filters
  + How UDP/TCP communication takes place
* **DIMY Notes**
  + Contact tracing
    - Use smartphone based digital contact tracing apps that exchance BLE messages to record contact.
    - Contact tracing app = Smartphones act as clients + backend server
      * When two individuals meet, smartphones exchange random identification code (when phones in close proximity)
      * Backend, maintained by health orgs, can request list of contacts on smartphone, when a user becomes diagnosed.
    - DIMY – the run down:
      * Connection between user devices established using Diffie-Hellman key exchange + shamir secret sharing (over insecure BLE broadcast channel).
      * Bloom filters used to store close contact info on both individual device and back end server.
        + Also, multiple close contacts on users phone stored in a SINGLE fixed-size bloom filter. The original contact info is then deleted, once encoded into bloom filter

Prevents info leakage at client level (e.g. theft) and back end

Reduced client/backend storage requirements

* + - * Blockchain based back-end design
        + Use of hashing algorithms to encrypt contact lists
    - Hybrid architecture
      * Server responsible for risk analysis/notification processing
      * Client manages generation of temp IDs (EphID/EncID)
* **Background Notes**
  + Diffie-Hellman
    - Uses elliptic curve multiplicative group
  + Shamir Secret Sharing
    - Making shares of a secret that can be securely distributed over many devices by a “threshold secret sharing mechanism”
    - Two phases: sharing & reconstruction
      * (k,n) secret sharing scheme means a secret S, shared among n people such that all n people receives parts of the secret, S, can be reconstructed by any group of k or more people. If k=n, all shares are necessary.
      * This scheme has to satisfy
        + Recoverability: secret can be constructed given any k shares
        + Secrecy: No info can be known about the secret given any shares < k.
  + Bloom filter
    - Probabilistic data structure, used to represent set membership (i.e. true or false).
    - When the BF is queries, it will return true if the data exists in the filter and false if not.
    - Implemented as bit array, BF[i]. Each of the n bits is accessed via h independent hash functions H1(x)…Hh(x). Querying x in the BF (i.e. checking the x exists in the BF) means checking that each of the h hash functions outputs 1 i.e.
    - 
    - This essentially means that querying “john” for example, with h = 3, we compute h1(john) = a, h2(john) = b, h3(john) = c. If BF[a] == BF[b] == BF[c] == 1, then we can say the john exists. Usually the hash functions are combined with a modulus to restrain the output to the size of the bit array.
  + Blockchain
    - Blocks chronologically sequential, immutable, linked to previous blocks using hash function.
    - Modelled after Hyperledger Fabric blockchain
* **Protocol Description**
  + Devices in DIMY, periodically generate **ephemeral identifiers**
    - Used in Diffie-Hellman to establish secret key, that would represent encounter between devices, if they come into contact
    - EphID(At) = g^(xAt), where xAt = random number at time, t, g is an element of G, elliptic curve group.
    - Then devices employ the k,n secret sharing scheme to produce n secret shares of the EphID. Devices then broadcast these shares, 1 share per minute, through BLE advert messages.
    - A device can reconstruct the EphID advertised from another device, if it has stayed in communication range of the broadcasting device for at least k minutes, as per k,n secret sharing protocol.
    - Lets say Alice reconstructs EphID(Bt) = g^(yBt), advertised by bob, where ybt is the random number generated by bob at time t. She can now get the secret encounter identifier, EncID(ABt) = (g^(xAt))^yAt. Bob can do the same, having received k advertisements from Alice.
  + **Encounter identifiers stored in Daily Bloom Filter**
    - Each device maintains a DBF, inserting all constructed EncIDs into it, for that day. The EncID is deleted as soon as it is entered into the DBF. After 21 days, the DBF is automatically deleted (i.e. 21 DBFs at one time max, constantly rotating)
  + **Infected users send 21 DBF to/query blockchain**
    - Once user is diagnosed with covid, device combines 21 DBF into one contact bloom filter, and uploads the CBF to the blockchain. CBF stored as transaction inside a block, which is appended to the blockchain.
    - Users who want to check if they came into close contact with a positive user query the blockchain daily.
      * Device combines all locally stored DBFs (max 21) into a single BF called a query bloom filter (QBF). The QBF as a query is uploaded to the block chain, which then matches the QBF with the CBF stored, as a transaction, on the blockchain and returns matched/unmatched
        + Matched = QBF stored separately for further verification by HA (health authorities)
        + Unmatched = QBF deleted.
* **In Depth**
  + Close contact representation
    - Shared secret representation to be used for contact representation (more resilience to replay attacks)
    - Diffie hellman used to share the secret among devices.
  + Generating identifiers
    - EphIDs valid for 30 mins
    - Size is 16 bytes (128 bits)
    - Devices don’t directly advertise the EphID, instead they use the k,n secret sharing mechanism.
  + Advertising & receiving identifiers
    - Device calculates n secret parts of the EphID, broadcasts each share at a rate of 1/min, after diffie-hellman used to encrypt the secret.
    - Reciever can reconstruct the EphID ONLY IF it has SUCCESSFULLY received k out of n shares.
      * Use k = 15, n = 30 (based on minimum duration of close contact being 15 mins by CDC)
    - Hash of ID also transmitted.
    - If k shares not received **OR** hash value fails integrity check, shares are discarded without attempting reconstruction.
    - If 10 chunks from EphID(1) + 10 chunks from EphID(2) received during rotation (i.e. total 20 minutes), contact wont be registered
      * This is fixed by broadcasting 2 EphIDs, rotating one identifier in such a way that the start of each EphID is staggered by 15 advertisment intervals.
      * Also include the hash of EphID, added to first 24 bits as random identifier (Use first 24 bits of SHA-256 output).
      * Once 15 shares received, reconstruct identifier, compute first 24 bits of sha-256 hash and confirm it is same as random identifier.
  + Storing encounter information
    - Ones EncID has been computed, (from secret key of both user devices)
    - They use 800000 bit array bloom field, with 2 hash functions
    - Low false positive rate (1/160400, when 1000 contacts in a day, input to DBF)
  + Uploading encounter identifiers to blockchain
    - Users can upload CBF (if infected) and QBF (to query blockchain and check for danger)
      * For CBF, last 21 DBFs are combined into a single CBF (size 100KB, same as DBF)
    - Hyperledger Fabric
      * HA maintains set of peer nodes that
        + Host legers
        + Execute smart contracts (e.g. handle query)
        + Maintain ordered set of nodes, for consensus protocol.
      * Smart contract used:
        + Issuing access tokens

Only users that have tested positive can upload to the blockchain.

This is done by HA giving them a temporary token, that gives them access to back end (permissioned blockchain platform)

Valid for 24 hours

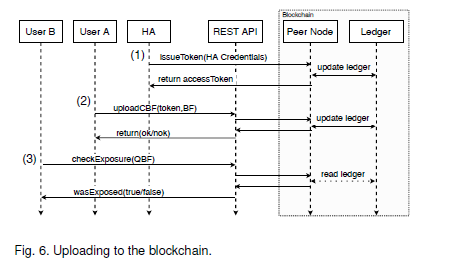
HA credentials given input to smart contract, output is: smart contract records token to blockchain.

* + - * + Processing CBF

After token is successfully validated, smart contract records CBF permanently to the block chain, and ledger states are updated

* + - * + Processing QBF

Smart contract compares users QBF to stored CBFs in the ledger. Smart contract then returns true/false depending on matches.



* + - CBFs stored in the ledgers on blockchain are managed/queried by Hyperledger.
      * On-chain data storage capacity is 4MB i.e. Hyperledger can add min: 1 and max: 40 CBFs (40x100KB = 4MB) in a single transaction
      * REST APIs used to ensure anonymity between user and blockchain (by use of a token granted by the HA, which acts as the authentication provider)
        + Web services – applications use diff web platforms (e.g. java, .net, node js)

This makes it difficult to ensure accurate communication between apps

Web services gives common platform to allow multiple apps built on diff programming languages to have ability to communicate with each other

SOAP and RESTFUL

* + - * + REST

Representational State Transfer

Set of rules devs follow when creating restful services

Works with http: get, post, put, delete requests

E.g. creating email drafts

* + - * + The REST APIs are provided by the HAs
      * Step 1 (No REST API here, direct backend)
        + HA issues temp access token by providing HA credentials to blockchain in transaction via peer node.
        + Peer node validates transaction, logs to blockchain (adds to ledger)
        + Token then returned to HA, which is then transmitted to user.
      * Step 2 (CBF, +ive covid case)
        + After receiving token, user can upload CBF using REST API, with access token.
        + REST API then anonymously forwards request to backend, and responds to user with transaction status.
      * Step 3 (check for exposure)
        + User interacts with REST API, by issuing “checkExposure” message, including the QBF.
        + REST API forwards query to blockchain, smart contract returns true or false in form of “wasExposed” message.
  + Contact verification process
    - Done through smart contract on blockchain
    - The app combines all 21 DBFs into a single QBF
    - Oldest bloom filter date, T(old) appended in query
    - Blockchain runs a search on the ledgers for this query, trying tto match any QBF entry with existing CBFs entries in transactions on the ledger.
    - Search restricted to transactions following the T(old) date.
    - As soon as single encounter matched between CBF and QBF, search terminated
    - Blockchain returns appropriate response.

**WHAT TO WRITE FOR MID TERM SUBMISSION (1st April)**

The midterm report should include the group number, members name and zIDs, assignment diary that details the overall plan with assigned task for each group member, progress on the implementation and issues faced. You are also to submit a recorded presentation, maximum of 6 slides, that explains your understanding of the DIMY protocol and what each component achieves in the protocol. You should divide the presentation equally among the group members with each member presenting their part. These mid-term deliverables carry 5 marks (2 marks for report and 3 marks for the presentation).

**IMPLEMENTATION**

* ECDH **-** [ECDH Key Exchange - Examples - Practical Cryptography for Developers (nakov.com)](https://cryptobook.nakov.com/asymmetric-key-ciphers/ecdh-key-exchange-examples)
  + The random number is X\_a and X\_b. The letter g is a shared constant value (think back to Diffie-Hellman). Person A calculates g to the power of X\_a, which gives their ephemeral ID. LIkewise, Person B calculates g to the power of X\_b.
  + They send these to each other using K out of N secret sharing. After K minutes, both people will have each other's ephemeral ID. Person A raises Person B's ephemeral ID to the power of X\_a, which gives g^X\_b^X\_a, which is the Encounter ID. Likewise for Person B, they calculate g^X\_a^X\_b.
  + Overall, this will give the same encounter ID for both people.
  + Use curve = secp128r1 for 128 bit generation, will actually be 129 bits, first bit is for odd/even (i.e. in hex it will be 02/03 because 1000 =