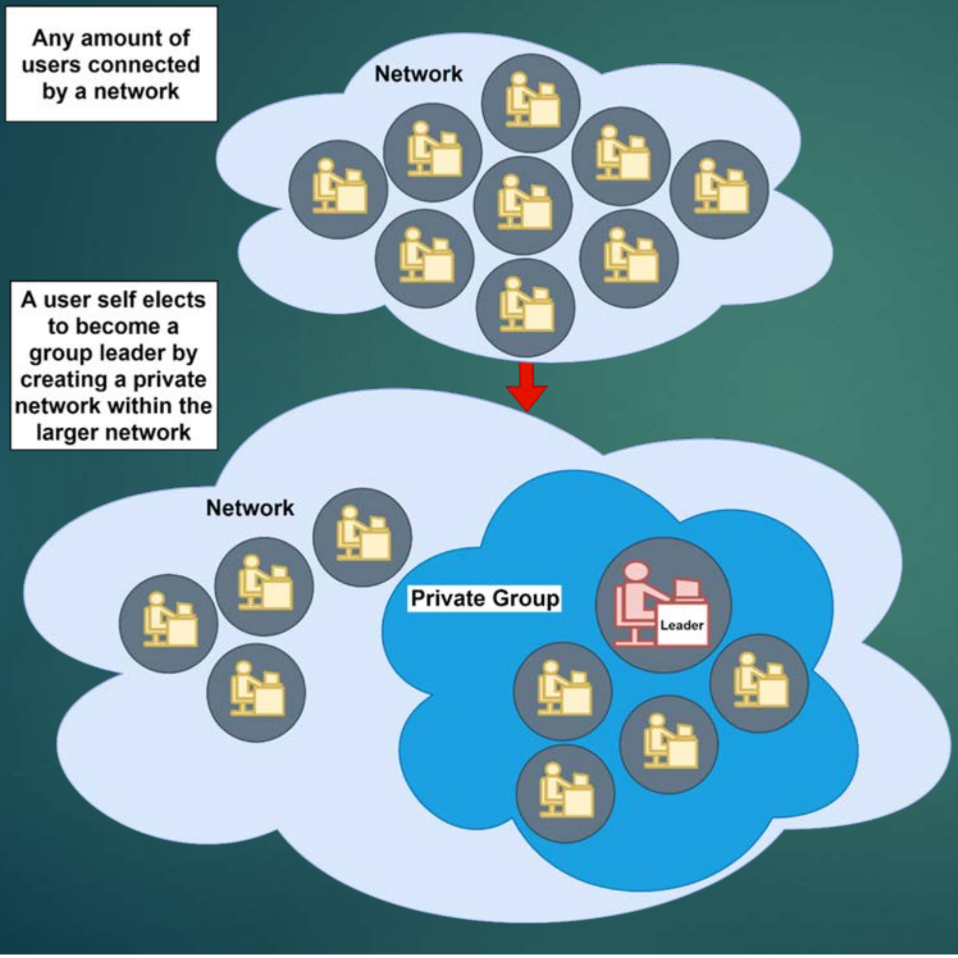
Private Group Communication in Blockchain Based on Diffie-Hellman Key Exchange

- Zachary Laney and **Yoohwan Kim**
- University of Nevada Las Vegas
- Yoohwan.Kim@unlv.edu

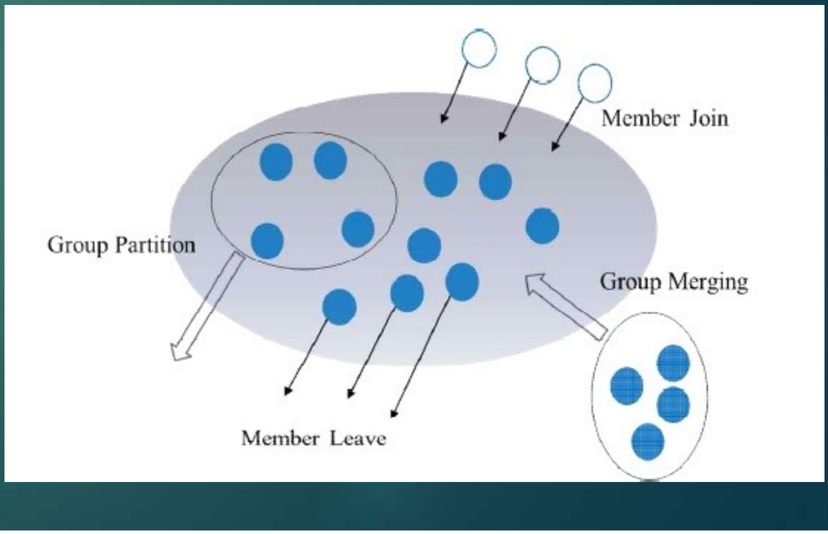
Private Group Communication



Any amount of users connected by a network

A user self elects to become a group leader by creating a private network within the larger network

Out of the total set of possible users with access to the open distributed blockchain any user may self elect to establish private group communication with any subset of the total amount of users using a **Public Key** and **Wallet Address**



Introduction

What is the problem with Private Group Communication?

Who has sent private information to another person on internet?

Who has sent private information to a group of people on the internet?

Who has sent private information to a group of people they've never met?

Is there a solution for securely sending private information to an anonymous group?

Current Group Key Protocols

What is the problem with the current standard of securely distributing a Group Key?

They require every user to actively participate in the entire Group Key creation process.

They require a trusted centralized third party to transfer the information.

Creation

Security

Delivery

The distributed immutability of blockchain technology can help.

Challenges

WITH ESTABLISHING PRIVATE GROUP COMMUNICATION ON THE OPEN DISTRIBUTED BLOCKCHAIN

Open Distributed Blockchain

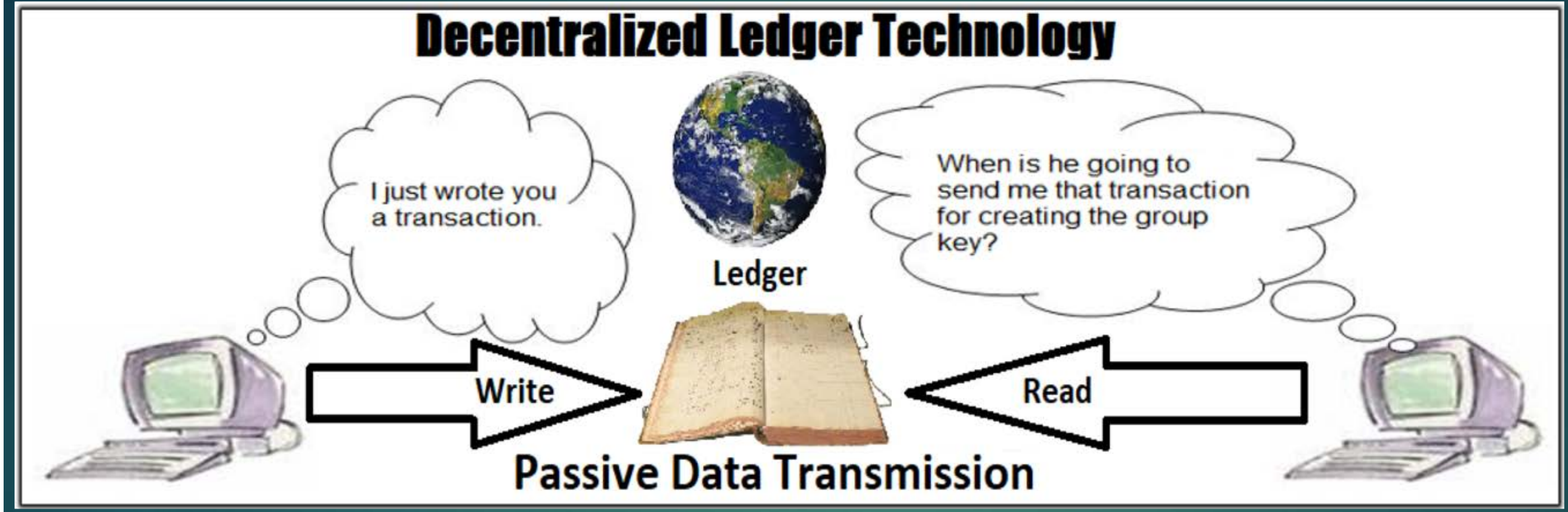
Three Main Challenges

Blockchain Attributes	Open Data		Closed Data	
Public Access	**Write Data**	Anyone	**Write Data**	Anyone
	Read Data	Anyone	**Read Data**	Permission
Private Access	**Write Data**	Permission	**Write Data**	Permission
	Read Data	Anyone	**Read Data**	Permission

In Ethereum anyone can read or write data.

Private Group Communication on the Open Distributed Blockchain

Challenge #1


Decentralized Ledger Technology
I just wrote you a transaction.
When is he going to send me that transaction for creating the group key?
Ledger
Write
Read
Passive Data Transmission

Passive

&

Non-interactive

- New information written to the ledger needs to notify group members.

Solution: Everyone transacts their information to a smart contract which creates a notification.

Private Group Communication on the Open Distributed Blockchain

Challenge #2

Distributed Immutability and **Public Verifiability** come with a cost.

Blockchain Attributes	Open Data	
Public Access	**Write Data**	Anyone
	Read Data	Anyone

Open Data

- Anyone can see the information on the ledger.

Solution: Encrypt the data.

Private Group Communication on the Open Distributed Blockchain

Challenge #3

Transferring information via the Open Distributed Blockchain **is inefficient**.

Slow Transactions

- The time for transactions takes anywhere from a second to an hour depending on the amount of gas provided and the current mining pool.

Expensive Transactions

- Smart Contracts aren't designed to be databases because of the high cryptocurrency costs for data storage to pay miners.

Permanent Transactions

- Encrypted data stored on the open distributed blockchain is permanent so if the key is stolen or brute forced at any point in the future it can be decrypted.

Solution #1: **Cope** with inefficiency.

Solution #2: Send information via **email**.

Diffie-Hellman Key Exchange

FOR ESTABLISHING A PRIVATE COMMUNICATION CHANNEL

Diffie- Hellman key Exchange Principle

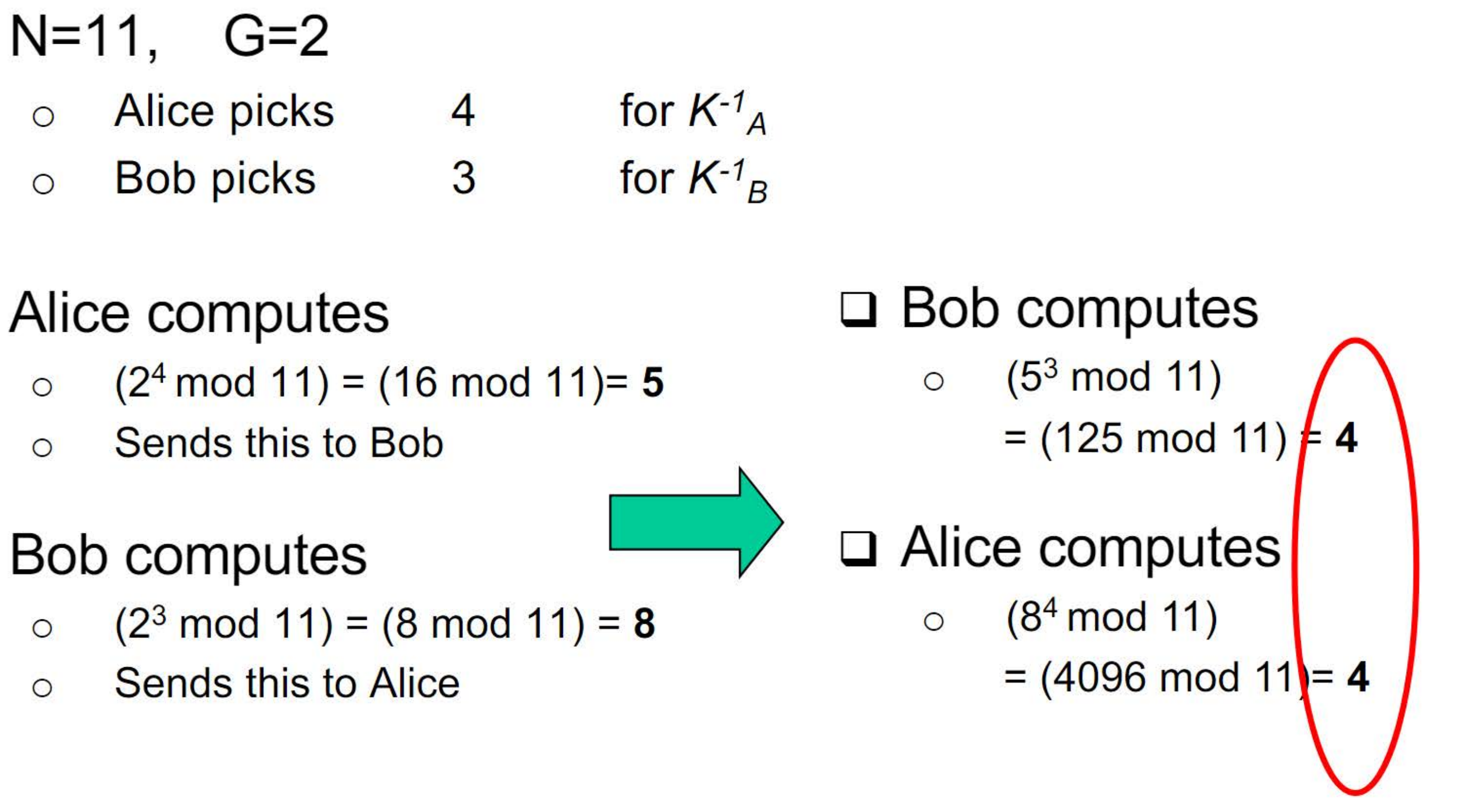
$$(g^x)^y = g^{xy} = (g^y)^x$$

$$K = (\underline{G^X \text{ mod } N})^Y \text{ mod } N$$

$$= (\underline{G^Y \text{ mod } N})^X \text{ mod } N$$

$$= \mathbf{G^{XY}} \text{ mod } N$$

Diffie- Hellman key Exchange Example



N=11, G=2

- ○ Alice picks 4 for K^{-1}_A
- ○ Bob picks 3 for K^{-1}_B

Alice computes

- ○ $(2^4 \bmod 11) = (16 \bmod 11) = \mathbf{5}$
- ○ Sends this to Bob

Bob computes

- ○ $(2^3 \bmod 11) = (8 \bmod 11) = \mathbf{8}$
- ○ Sends this to Alice

❑ Bob computes

- ○ $(5^3 \bmod 11)$
  $= (125 \bmod 11) = \mathbf{4}$

❑ Alice computes

- ○ $(8^4 \bmod 11)$
  $= (4096 \bmod 11) = \mathbf{4}$

Previous Research

FOR ESTABLISHING PRIVATE GROUP COMMUNICATION

Group Key Creation with Diffie- Hellman key Exchange

$$((g^{x})^{y})^{z} = g^{xyz} = (g^{y})^{x})^{z}$$

$$K = ((\underline{G^{X} \bmod N})^{Y} \bmod N)^{Z} \bmod N$$

$$= ((\underline{G^{Y} \bmod N})^{Z} \bmod N)^{X} \bmod N$$

$$= ((\underline{G^{Z} \bmod N})^{X} \bmod N)^{Y} \bmod N$$

$$= \mathbf{G}^{\mathbf{XYZ}} \bmod N$$

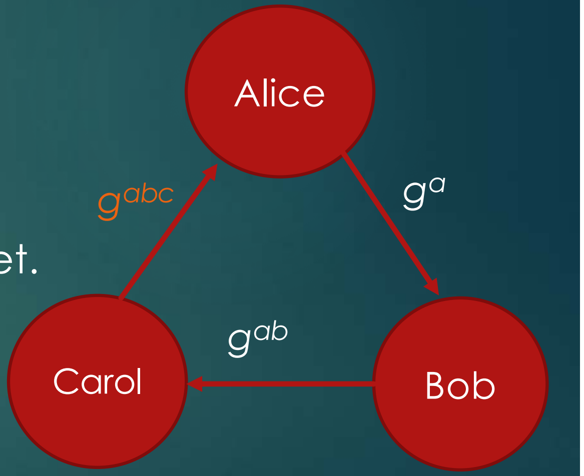
Shared key among multiple parties

- Group key creation among 3 members
  - Alice computes g^a and sends it to Bob.
  - Bob computes $(g^a)^b = g^{ab}$ and sends it to Carol.
  - Carol computes $(g^{ab})^c = g^{abc}$ and uses it as her secret.

  - Bob computes g^b and sends it to Carol.
  - Carol computes $(g^b)^c = g^{bc}$ and sends it to Alice.
  - Alice computes $(g^{bc})^a = g^{bca} = g^{abc}$ and uses it as her secret.

  - Carol computes g^c and sends it to Alice.
  - Alice computes $(g^c)^a = g^{ca}$ and sends it to Bob.
  - Bob computes $(g^{ca})^b = g^{cab} = g^{abc}$ and uses it as his secret.

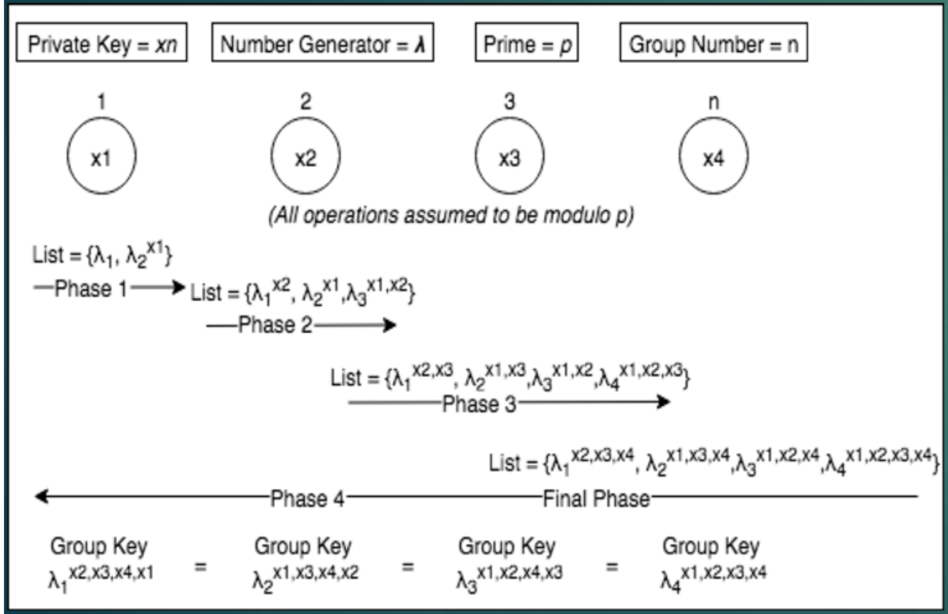
  - *Problem: Everybody must do (n) discrete exponentiation*



Group Diffie Hellman Key Exchange Algorithm Based Secure Group Communication

Lavanya R, Dr. S V Sathyanarayana.
International Conference on Applied and Theoretical Computing and Communication Technology, IEEE 2017

Ring Diagram


Private Key = *xn*
Number Generator = λ
Prime = *p*
Group Number = n
1
2
3
n
x1
x2
x3
x4
(All operations assumed to be modulo p)
List = $\{\lambda_1, \lambda_2^{x1}\}$
Phase 1
List = $\{\lambda_1^{x2}, \lambda_2^{x1}, \lambda_3^{x1,x2}\}$
Phase 2
List = $\{\lambda_1^{x2,x3}, \lambda_2^{x1,x3}, \lambda_3^{x1,x2}, \lambda_4^{x1,x2,x3}\}$
Phase 3
List = $\{\lambda_1^{x2,x3,x4}, \lambda_2^{x1,x3,x4}, \lambda_3^{x1,x2,x4}, \lambda_4^{x1,x2,x3,x4}\}$
Phase 4
Final Phase
Group Key $\lambda_1^{x2,x3,x4,x1}$ = Group Key $\lambda_2^{x1,x3,x4,x2}$ = Group Key $\lambda_3^{x1,x2,x4,x3}$ = Group Key $\lambda_4^{x1,x2,x3,x4}$

Exponentiations: *n*

How it works

Up Flow Stage

- User 1 adds to a list every possible combination of the generator raised to the power of their private key and passes it to User 2.
- User 2 adds to a list every possible combination of the generator raised to the power of their private key and passes it to User 3.
- User 3 adds to a list every possible combination of the generator raised to the power of their private key and passes it to User 4.
- The last user raises every number in list to the power of their private key.

Down Flow Stage

- The last user sends the list left back to User 3, User 2, User 1 who then raise their group key to the power of their private key.
- User 1, User 2, User 3, User 4 have calculated the same group key.

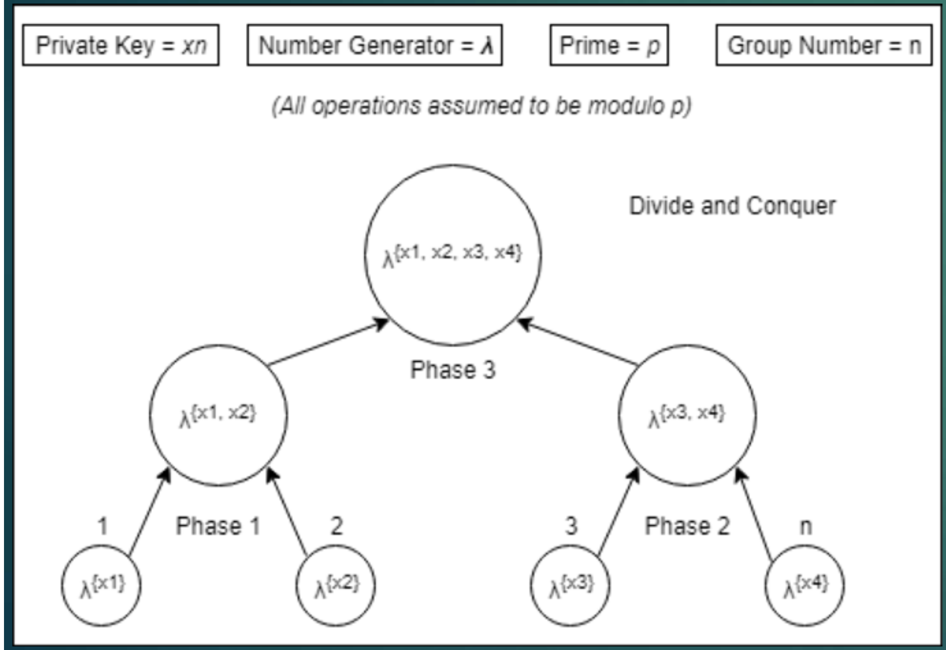
A better way? (e.g., 8 people)

- With divide-and-conquer, **A, B, C, and D** each perform one exponentiation, yielding g^{abcd}; this value is sent to **E, F, G, and H**.
    - In return, participants A, B, C, and D receive $\boldsymbol{g^{efgh}}$.
  - **A and B** each perform one exponentiation, yielding $\boldsymbol{g^{efghab}}$, which they send to **C and D**, while C and D do the same, yielding $\boldsymbol{g^{efghcd}}$, which they send to A and B.
  - **A** performs an exponentiation, yielding $\boldsymbol{g^{efghcda}}$, which it sends to B; similarly, B sends $\boldsymbol{g^{efghcdb}}$ to A. C and D do similarly.
  - **A** performs one final exponentiation, yielding $g^{efghcdba} = g^{abcdefgh}$, while B does the same to get $g^{efghcdab} = g^{abcdefgh}$; again, C and D do similarly.
  - **E through H** simultaneously perform the same operations starting g^{abcd}
  - Once completed, all participants will possess the secret $\boldsymbol{g^{abcdefgh}}$, but each participant will have performed **only four** modular exponentiations, **not eight** by a simple circular arrangement.
- We can reduce the number of modular exponentiations to **$\log_2(N) + 1$**

An Efficient Improved Group Key Agreement Protocol Based on Diffie-Hellman Key Exchange

Yang Guang-ming, Lu Ya-feng, MA Da-ming.
IEEE, 2017

Binary Tree Diagram



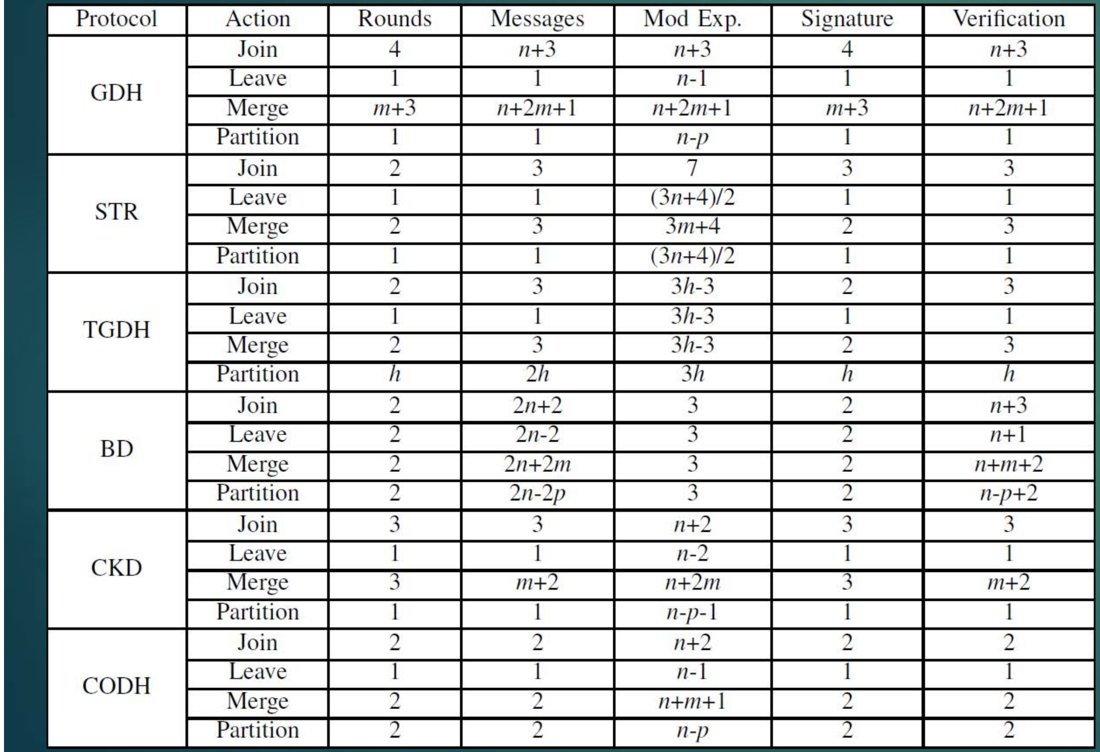
Exponentiations: $log_2(n) + 1$

How it works

- User 1 and User 2 both raise the generator to the power of their private keys and trade with each other.
- User 2 raises User 1's result to the power of their private key.
- User 1 raises User 2's result to the power of their private key.
- User 3 and 4 perform the same steps as User 1 and User 2.
- User 1 trades their result with User 3 and raises the other's result with their current number.
- User 2 trades their result with User 4 and raises the other's result with their current number.
- User 1, User 2, User 3, and User 4 have all calculated the same group key.

Secure Collaborative Key Management for Dynamic Groups in Mobile Networks

C. J. a. M. H. Sukin Kang
Journal of Applied Mathematics, 2014.



Protocol	Action	Rounds	Messages	Mod Exp.	Signature	Verification
GDH	Join	4	n+3	n+3	4	n+3
	Leave	1	1	n-1	1	1
	Merge	m+3	n+2m+1	n+2m+1	m+3	n+2m+1
	Partition	1	1	n-p	1	1
STR	Join	2	3	7	3	3
	Leave	1	1	(3n+4)/2	1	1
	Merge	2	3	3m+4	2	3
	Partition	1	1	(3n+4)/2	1	1
TGDH	Join	2	3	3h-3	2	3
	Leave	1	1	3h-3	1	1
	Merge	2	3	3h-3	2	3
	Partition	h	2h	3h	h	h
BD	Join	2	2n+2	3	2	n+3
	Leave	2	2n-2	3	2	n+1
	Merge	2	2n+2m	3	2	n+m+2
	Partition	2	2n-2p	3	2	n-p+2
CKD	Join	3	3	n+2	3	3
	Leave	1	1	n-2	1	1
	Merge	3	m+2	n+2m	3	m+2
	Partition	1	1	n-p-1	1	1
CODH	Join	2	2	n+2	2	2
	Leave	1	1	n-1	1	1
	Merge	2	2	n+m+1	2	2
	Partition	2	2	n-p	2	2

Group Diffie-Hellman (GDH)

Skinny Tree (STR)

Tree-Based Group Diffie-Hellman (TGDH)

Burmester and Desmedt (BD)

Centralized Key Distribution (CKD)

Collaborative Diffie-Hellman (CDH)

Five Main Components of Group Key Protocols:

Size of message

Amount of messages

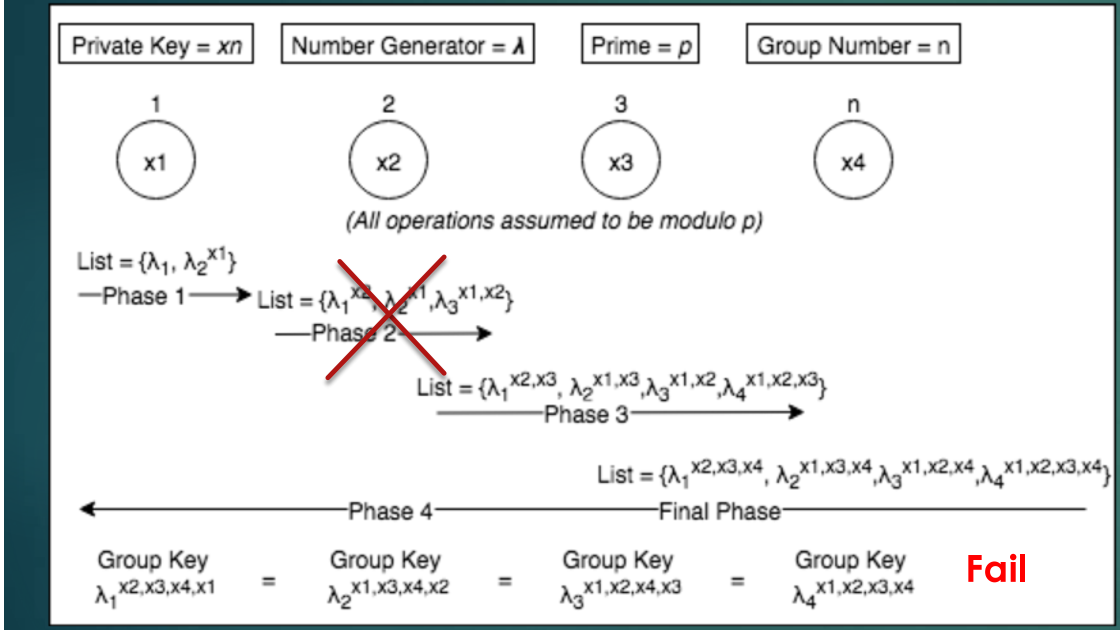
Amount of exponentiations

Data Structure

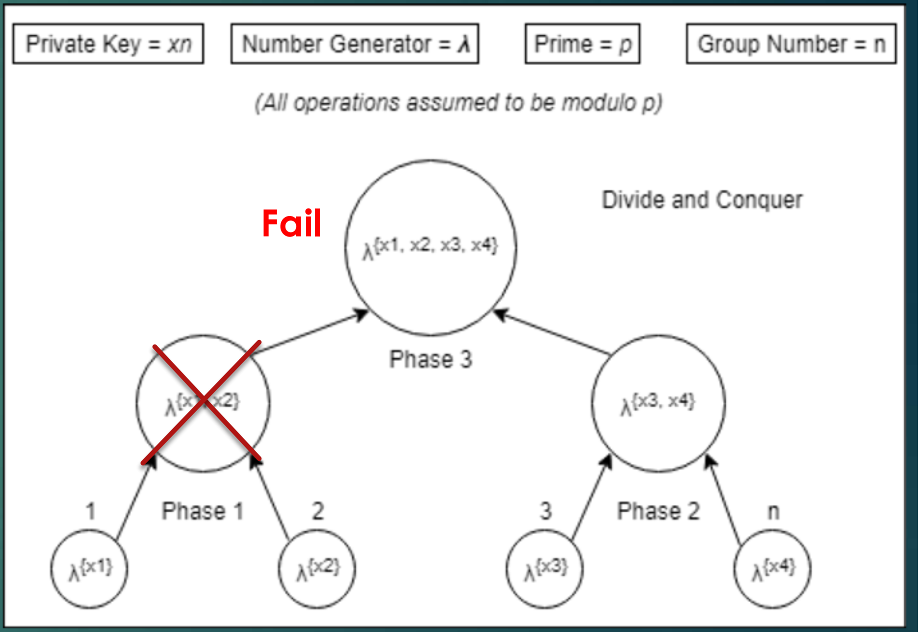
Balance of operations between the User and the Group Leader

Problem #1 with Traditional Group Key Algorithms - Needs an interaction

Ring


Private Key = *xn*
Number Generator = λ
Prime = *p*
Group Number = n
1
2
3
n
x1
x2
x3
x4
(All operations assumed to be modulo p)
List = $\{\lambda_1, \lambda_2^{x1}\}$
Phase 1
List = $\{\lambda_1^{x2}, \lambda_2^{x1}, \lambda_3^{x1,x2}\}$
Phase 2
List = $\{\lambda_1^{x2,x3}, \lambda_2^{x1,x3}, \lambda_3^{x1,x2}, \lambda_4^{x1,x2,x3}\}$
Phase 3
List = $\{\lambda_1^{x2,x3,x4}, \lambda_2^{x1,x3,x4}, \lambda_3^{x1,x2,x4}, \lambda_4^{x1,x2,x3,x4}\}$
Phase 4
Final Phase
Group Key $\lambda_1^{x2,x3,x4,x1}$ = Group Key $\lambda_2^{x1,x3,x4,x2}$ = Group Key $\lambda_3^{x1,x2,x4,x3}$ = Group Key $\lambda_4^{x1,x2,x3,x4}$
Fail

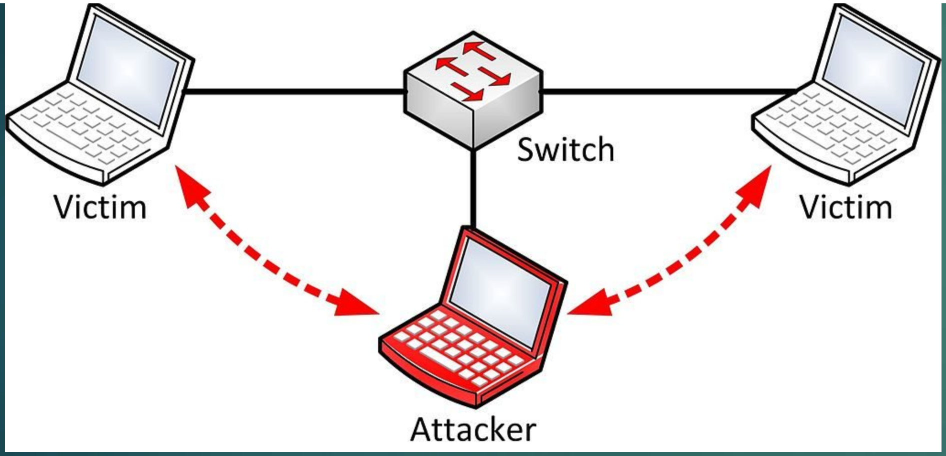
Binary Tree



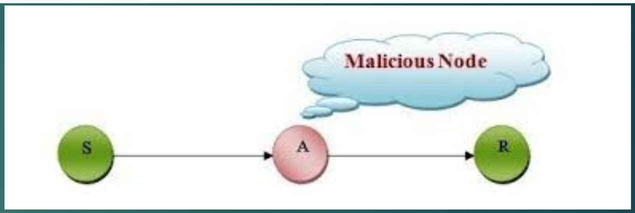
All users must actively participate in all phases of the process.

Problem #2 with Traditional Group Key Algorithms

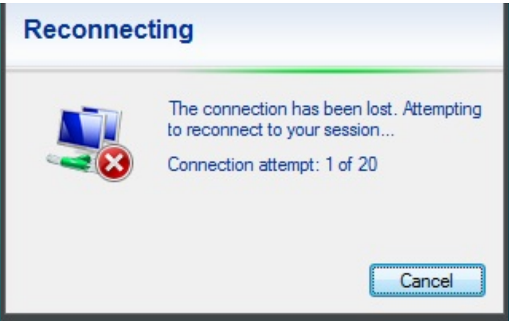
Man in the Middle Attack



Impersonation Attack



Service Outage


Reconnecting

The connection has been lost. Attempting to reconnect to your session...

Connection attempt: 1 of 20

Proposed System

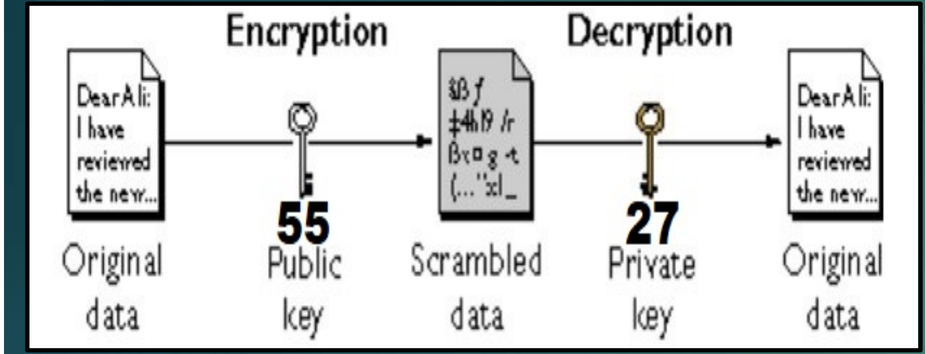
FOR ESTABLISHING PRIVATE GROUP COMMUNICATION
ON THE OPEN DISTRIBUTED BLOCKCHAIN

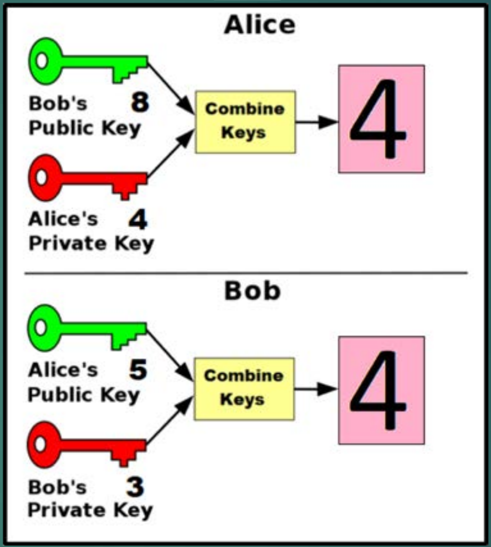
Private Keys, Public Keys, and Wallet Addresses

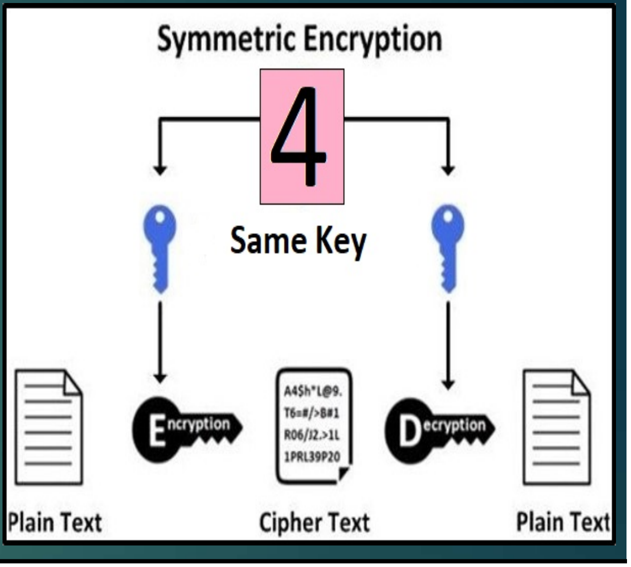
How are Private and Public Keys used? How are Wallet Addresses created?

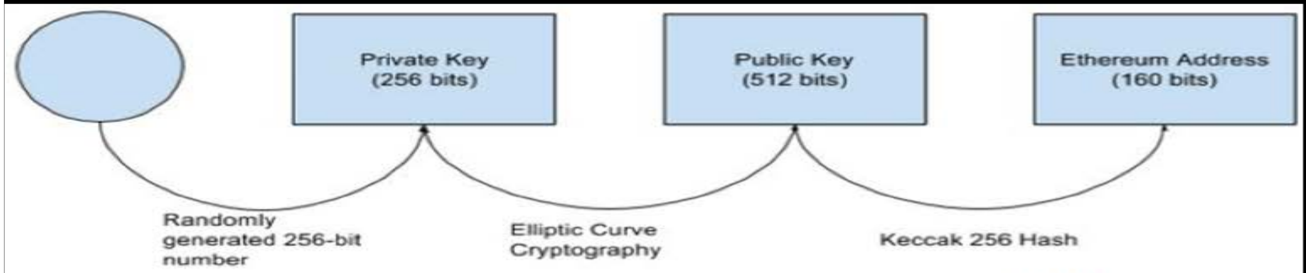
There are three types of encryption:

1. → **2.** → **3.**








Private Key (256 bits)
Public Key (512 bits)
Ethereum Address (160 bits)
Randomly generated 256-bit number
Elliptic Curve Cryptography
Keccak 256 Hash
Ethereum

A Wallet Address is created from Public Key #1

Private Group Communication on the Open Distributed Blockchain

Our Proposed Solution. But not the only solution available.

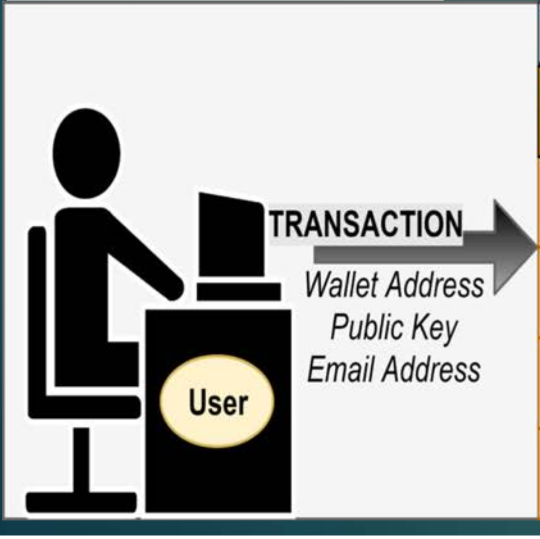
A Private Group Communication Channel is established and utilized outside of the blockchain using **EMAIL.**

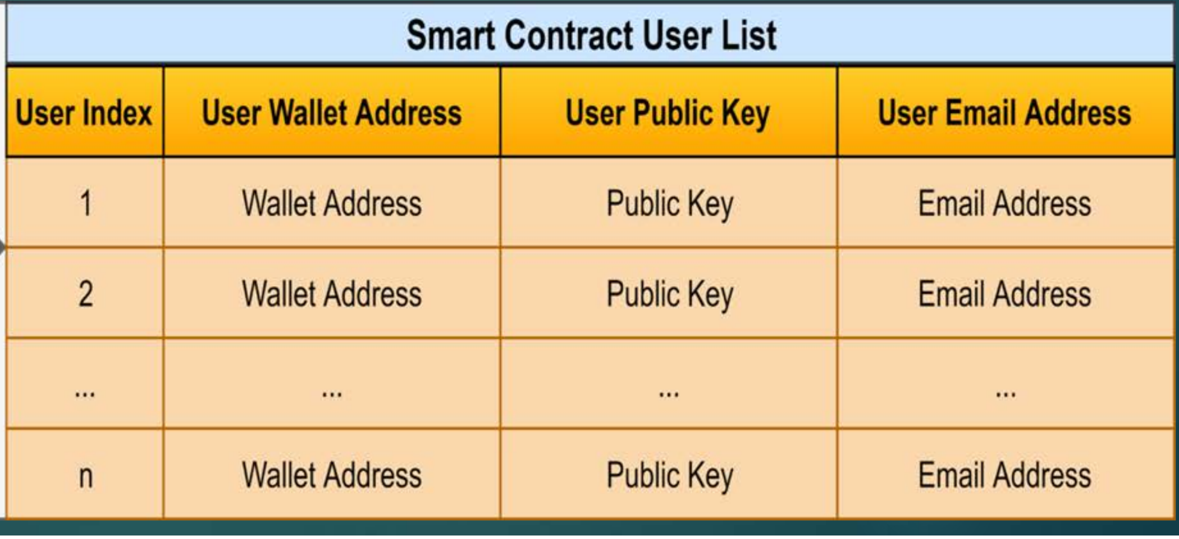
Requires all users to *transact* **3** items to a smart contract

Wallet Address

Public Key

Email Address



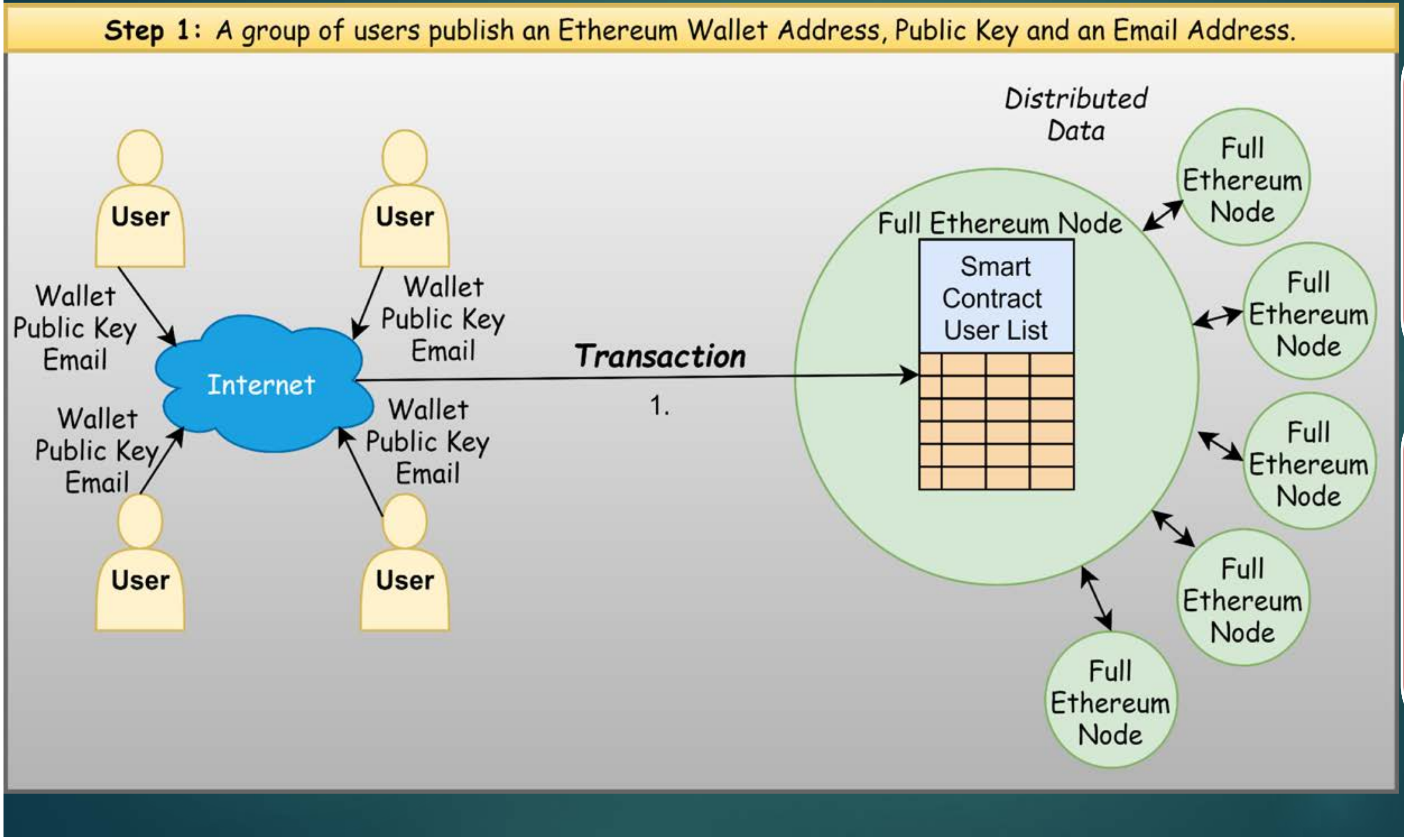


Smart Contract User List

User Index	User Wallet Address	User Public Key	User Email Address
1	Wallet Address	Public Key	Email Address
2	Wallet Address	Public Key	Email Address
...	...	...	...
n	Wallet Address	Public Key	Email Address

Private Group Communication on the Open Distributed Blockchain

Solution: **Step 1**



Step 1: A group of users publish an Ethereum Wallet Address, Public Key and an Email Address.

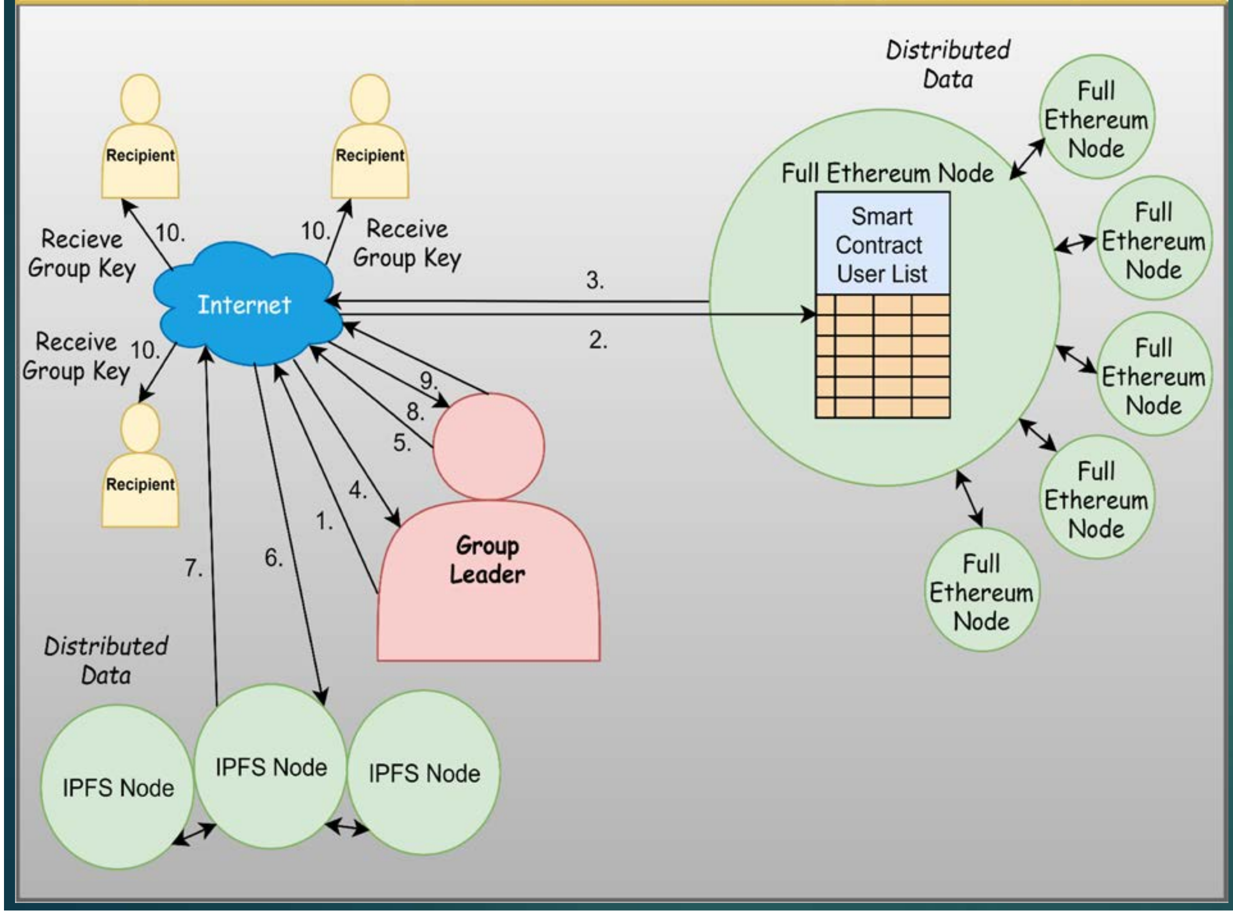
The **Wallet Address, Public Key, and Email** are permanently published to the open blockchain

Other users may then retrieve these items to establish private group communication by calculating a shared key to encrypt a Group Key in **Step 2** by using Diffie-Hellman cryptography

Private Group Communication on the Open Distributed Blockchain

Solution: **Step 2**

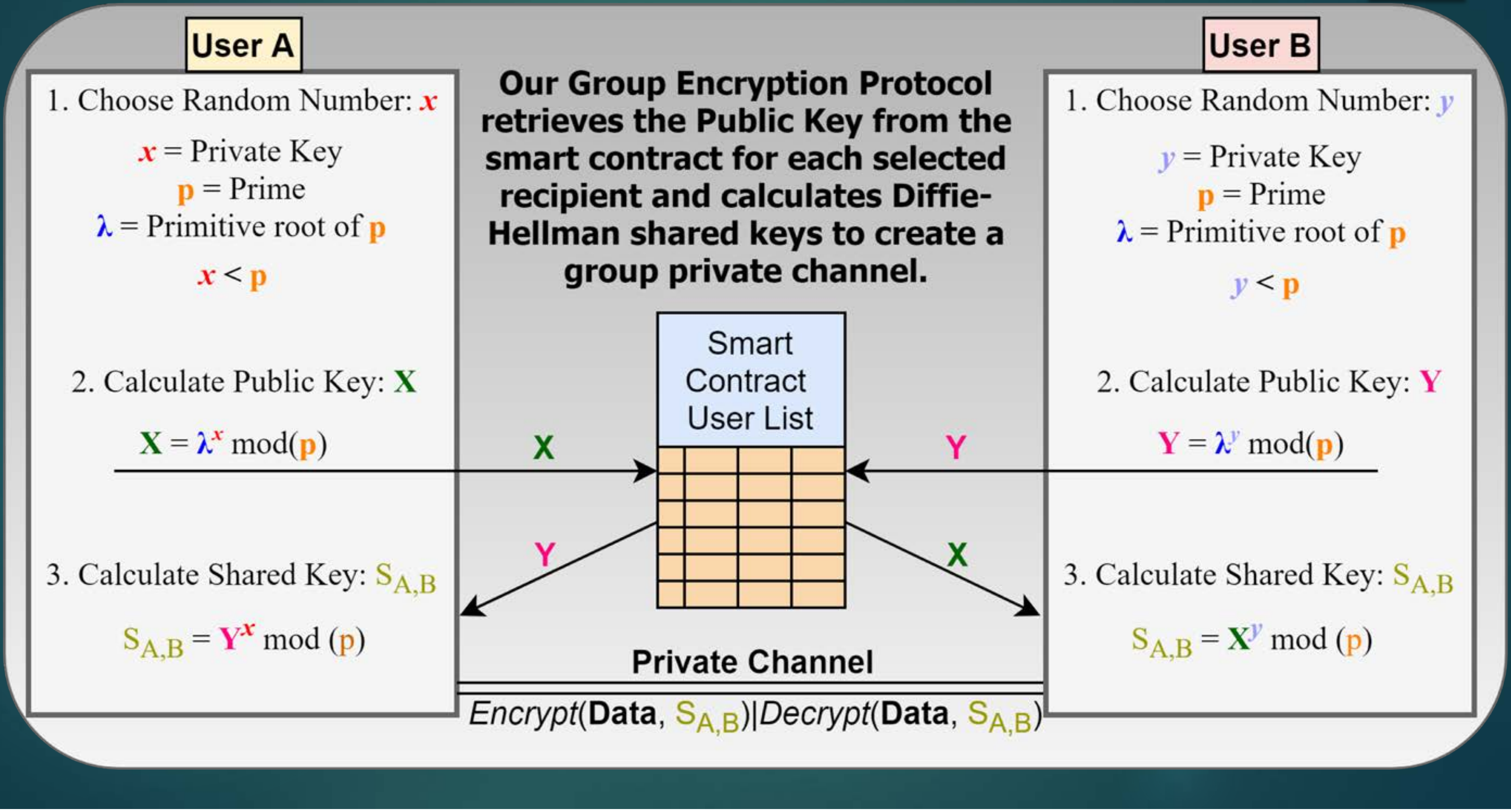
Step 2: A self elected Group Leader creates and distributes a Group Key to selected recipients.



1. Group Leader requests the user list from a Web Interface.
2. Web Interface requests the user list from a Smart Contract.
3. Smart Contract sends the user list back to the Web Interface.
4. The Group Leader receives the user list and builds the *GroupJSONFile*.
5. The Group Leader uploads the *GroupJSONFile* to IPFS.
6. IPFS receives the file and distributes it to a data hosting node.
7. IPFS returns the hash string to download the *GroupJSONFile*.
8. Group Leader receives the IPFS hash string.
9. Group Leader emails the JSON extractor web link and IPFS hash string to selected recipients.
10. Recipients receive the *GroupJSONFile* IPFS hash string and JSON Extractor web link to extract the Group Key.

Private Group Communication on the Open Distributed Blockchain

Pairwise Diffie-Hellman Cryptography



User A

1. Choose Random Number: x

x = Private Key

p = Prime

λ = Primitive root of p

$x < p$

2. Calculate Public Key: X

$X = \lambda^{x} \bmod(p)$

3. Calculate Shared Key: $S_{A,B}$

$S_{A,B} = Y^{x} \bmod (p)$

Our Group Encryption Protocol retrieves the Public Key from the smart contract for each selected recipient and calculates Diffie-Hellman shared keys to create a group private channel.

Smart Contract User List

Private Channel

Encrypt(**Data**, $S_{A,B}$)|*Decrypt*(**Data**, $S_{A,B}$)

User B

1. Choose Random Number: y

y = Private Key

p = Prime

λ = Primitive root of p

$y < p$

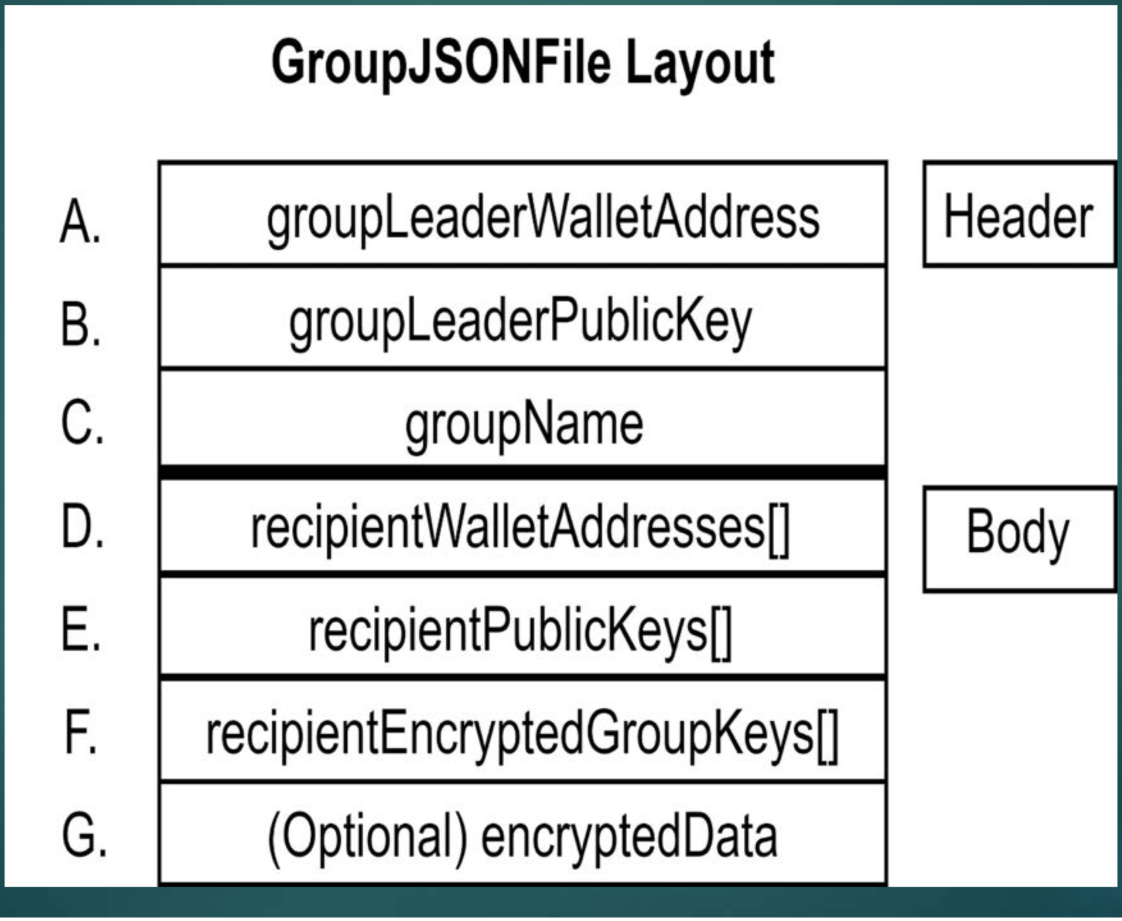
2. Calculate Public Key: Y

$Y = \lambda^{y} \bmod(p)$

3. Calculate Shared Key: $S_{A,B}$

$S_{A,B} = X^{y} \bmod (p)$

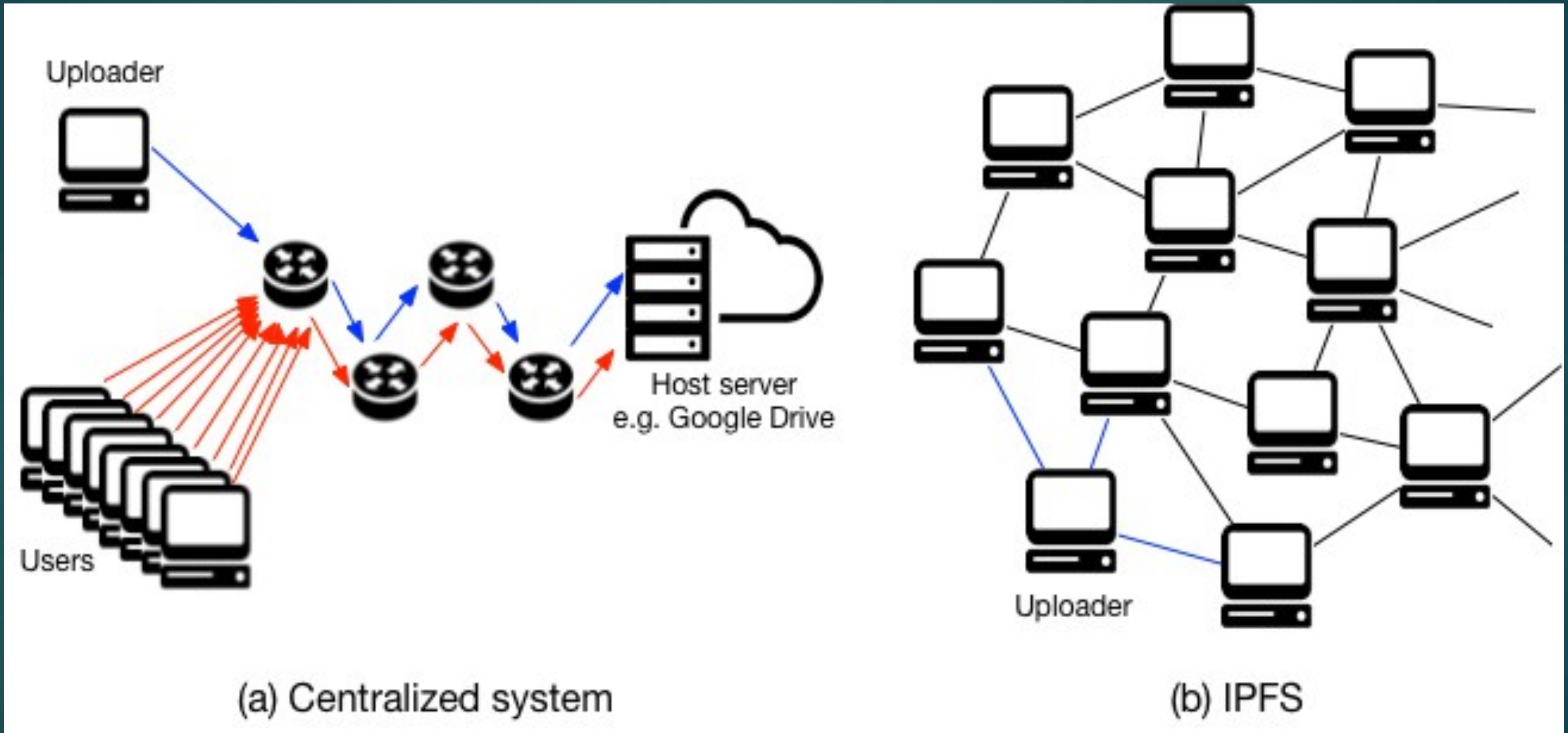
GroupJSONFile Architectural Layout



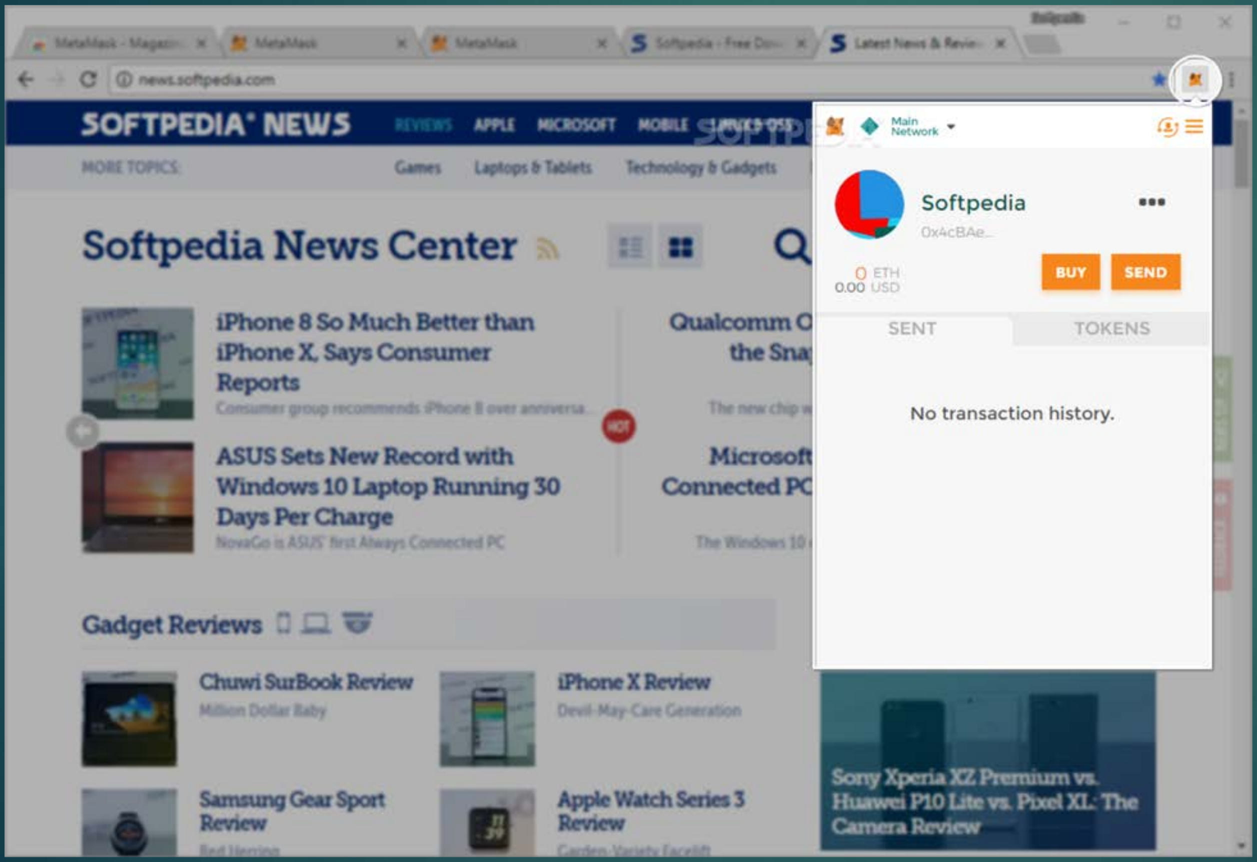
GroupJSONFile Layout

A.	groupLeaderWalletAddress	Header
B.	groupLeaderPublicKey	
C.	groupName	
D.	recipientWalletAddresses[]	Body
E.	recipientPublicKeys[]	
F.	recipientEncryptedGroupKeys[]	
G.	(Optional) encryptedData	

Interplanetary File System (IPFS)



MetaMask

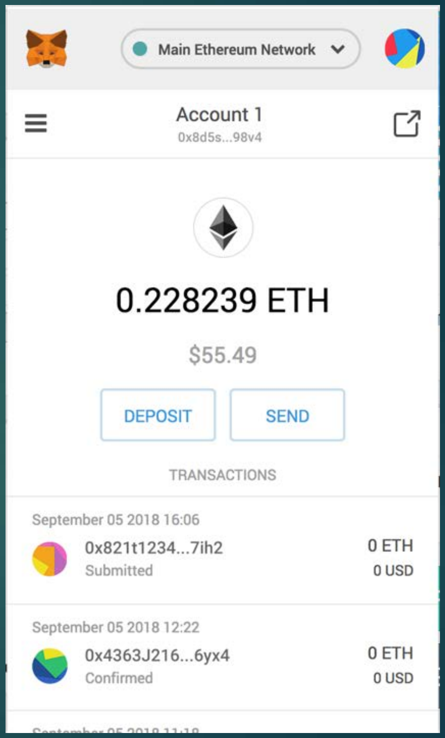

news.softpedia.com
SOFTPEDIA® NEWS
REVIEWS
APPLE
MICROSOFT
MOBILE
MORE TOPICS:
Games
Laptops & Tablets
Technology & Gadgets
Softpedia News Center
iPhone 8 So Much Better than iPhone X, Says Consumer Reports
ASUS Sets New Record with Windows 10 Laptop Running 30 Days Per Charge
Gadget Reviews
Chuwi SurBook Review
iPhone X Review
Samsung Gear Sport Review
Apple Watch Series 3 Review
Sony Xperia XZ Premium vs. Huawei P10 Lite vs. Pixel XL: The Camera Review
Main Network
BUY
SEND
SENT
TOKENS

Softpedia

0x4cBAe...

0 ETH
0.00 USD

No transaction history.



Account 1
0x8d5s...98v4

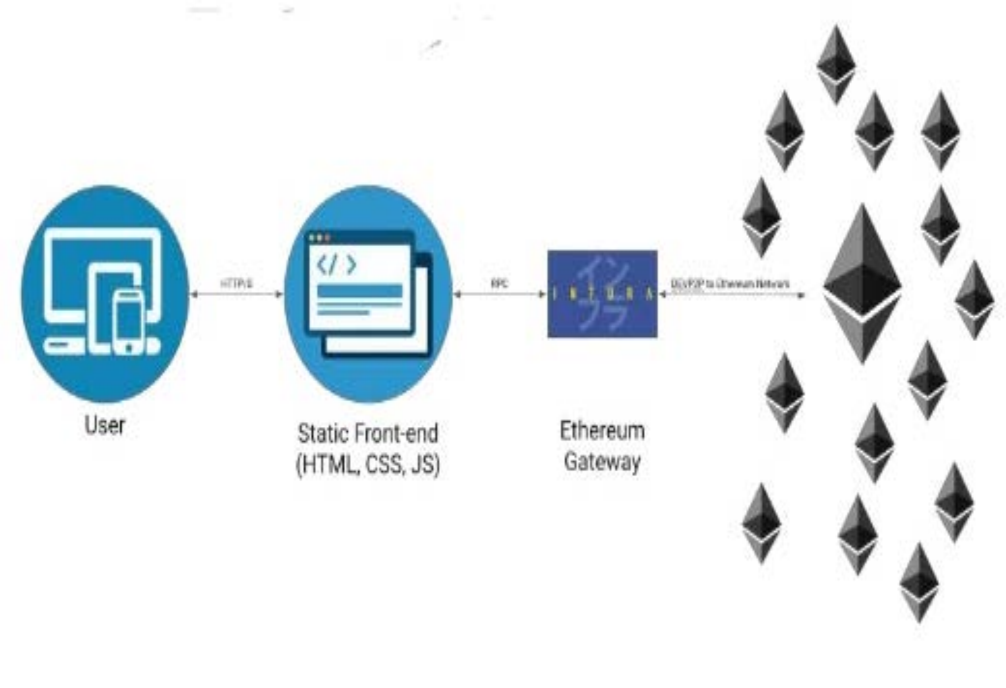
0.228239 ETH

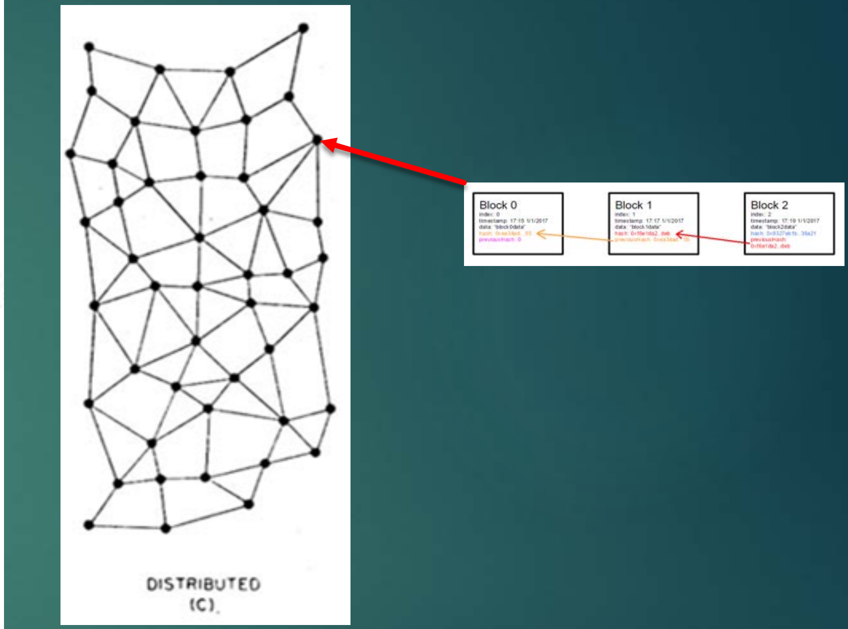
$55.49

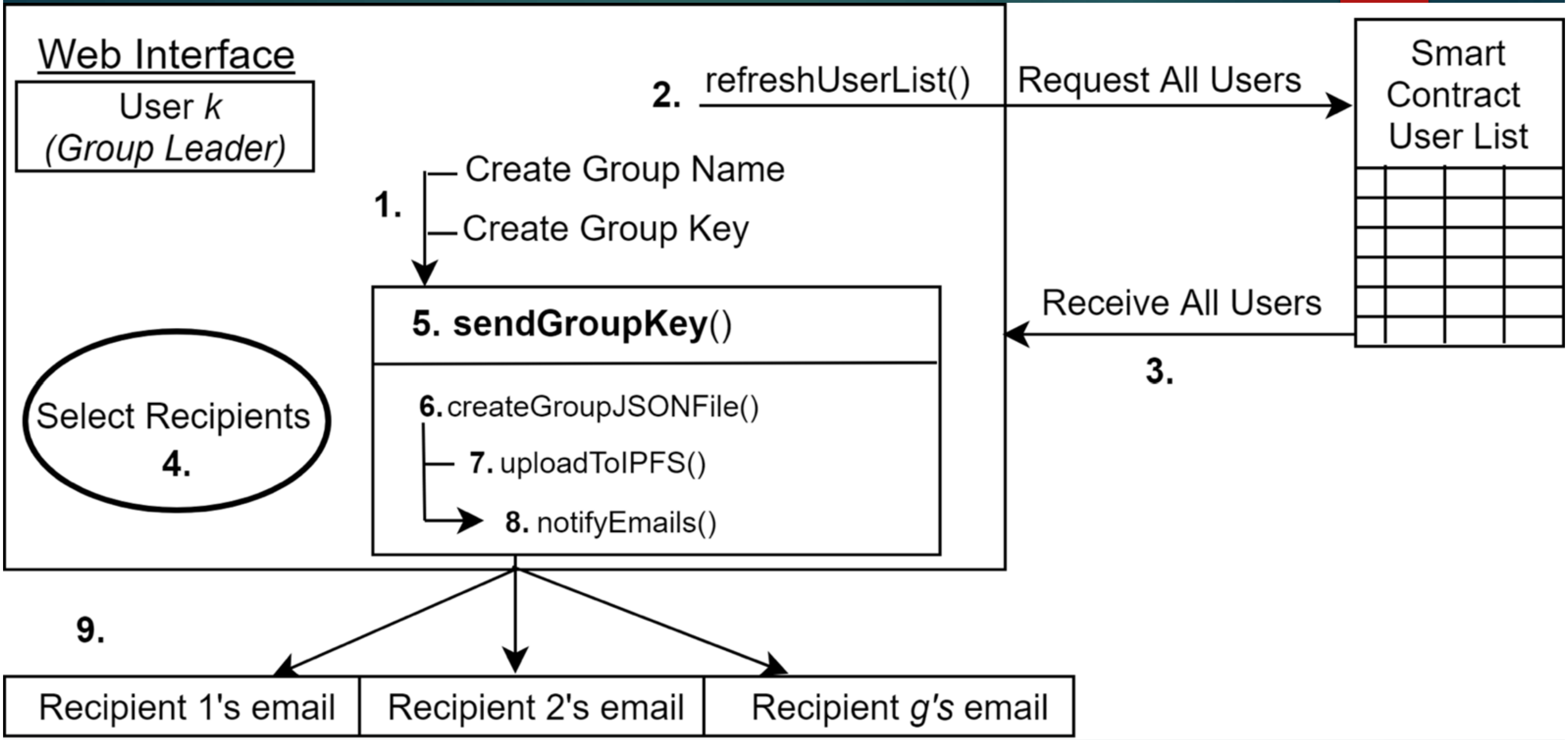
September 05 2018 16:06
0x821t1234...7ih2 — 0 ETH
Submitted — 0 USD

September 05 2018 12:22
0x4363J216...6yx4 — 0 ETH
Confirmed — 0 USD

Infura


User
HTTP/S
Static Front-end
(HTML, CSS, JS)
RPC
INFURA
Ethereum
Gateway
DEVP2P to Ethereum Network


DISTRIBUTED
(C)
Block 0
index: 0
timestamp: 17:15 1/1/2017
data: "block0data"
hash: 0xea34ad...55
previousHash: 0
Block 1
index: 1
timestamp: 17:17 1/1/2017
data: "block1data"
hash: 0xf6e1da2...deb
previousHash: 0xea34ad...55
Block 2
index: 2
timestamp: 17:19 1/1/2017
data: "block2data"
hash: 0x9327eb1b...36a21
previousHash:
0xf6e1da2...deb



Web Interface

User *k*
(Group Leader)

1. Create Group Name

Create Group Key

2. refreshUserList() Request All Users

Smart Contract User List

Receive All Users

3.

4. Select Recipients

5. sendGroupKey()

6. createGroupJSONFile()

7. uploadToIPFS()

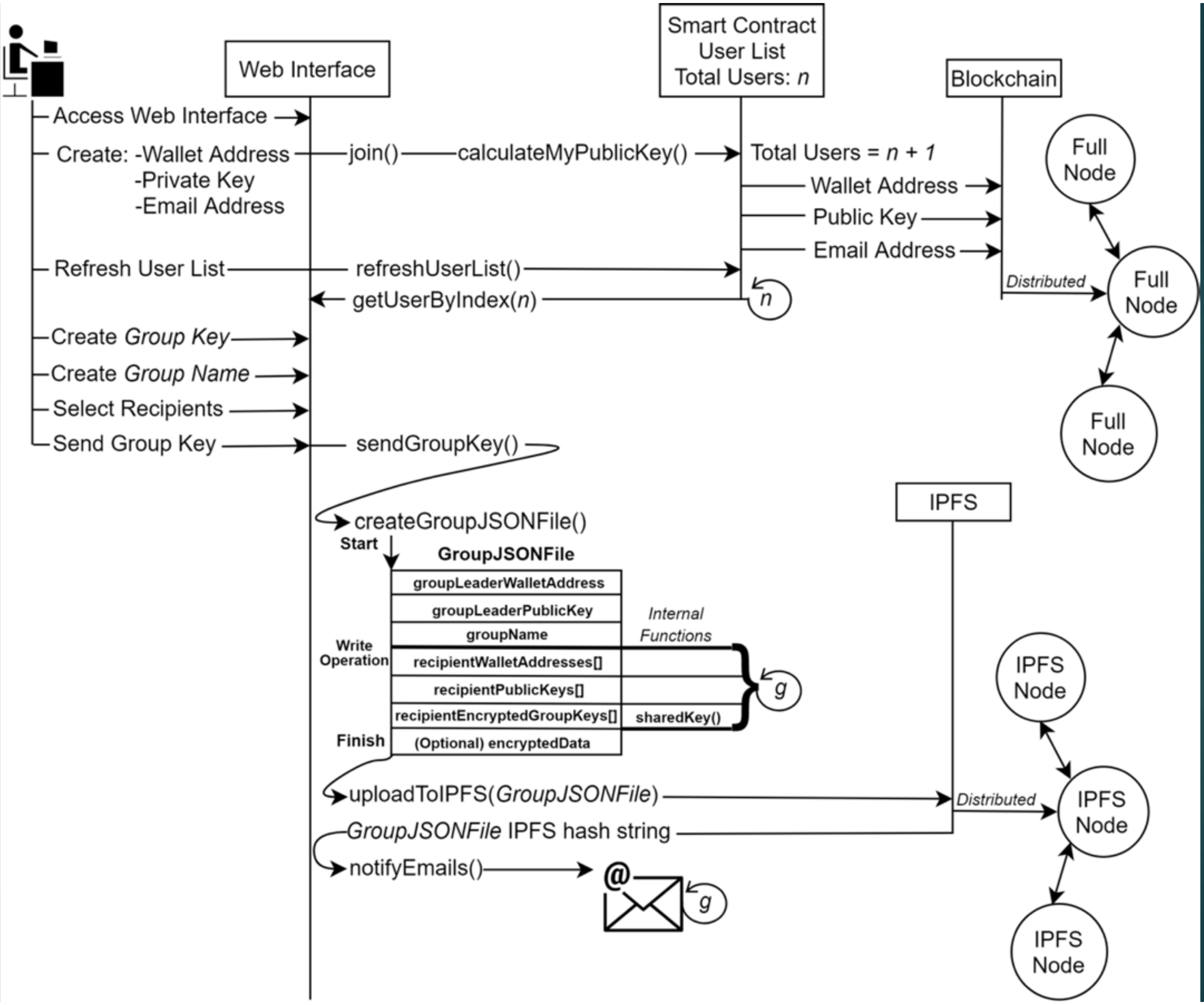
8. notifyEmails()

9.

Recipient 1's email

Recipient 2's email

Recipient *g's* email



Web Interface

Smart Contract
User List
Total Users: *n*

Blockchain

Access Web Interface

Create: -Wallet Address
-Private Key
-Email Address

join()

calculateMyPublicKey()

Total Users = *n* + *1*

Wallet Address

Public Key

Email Address

Refresh User List

refreshUserList()

getUserByIndex(*n*)

n

Create *Group Key*

Create *Group Name*

Select Recipients

Send Group Key

sendGroupKey()

Distributed

Full Node

Full Node

Full Node

IPFS

createGroupJSONFile()

Start

GroupJSONFile

groupLeaderWalletAddress
groupLeaderPublicKey
groupName
recipientWalletAddresses[]
recipientPublicKeys[]
recipientEncryptedGroupKeys[]
(Optional) encryptedData

Write Operation

Finish

Internal Functions

sharedKey()

g

uploadToIPFS(*GroupJSONFile*)

Distributed

GroupJSONFile IPFS hash string

notifyEmails()

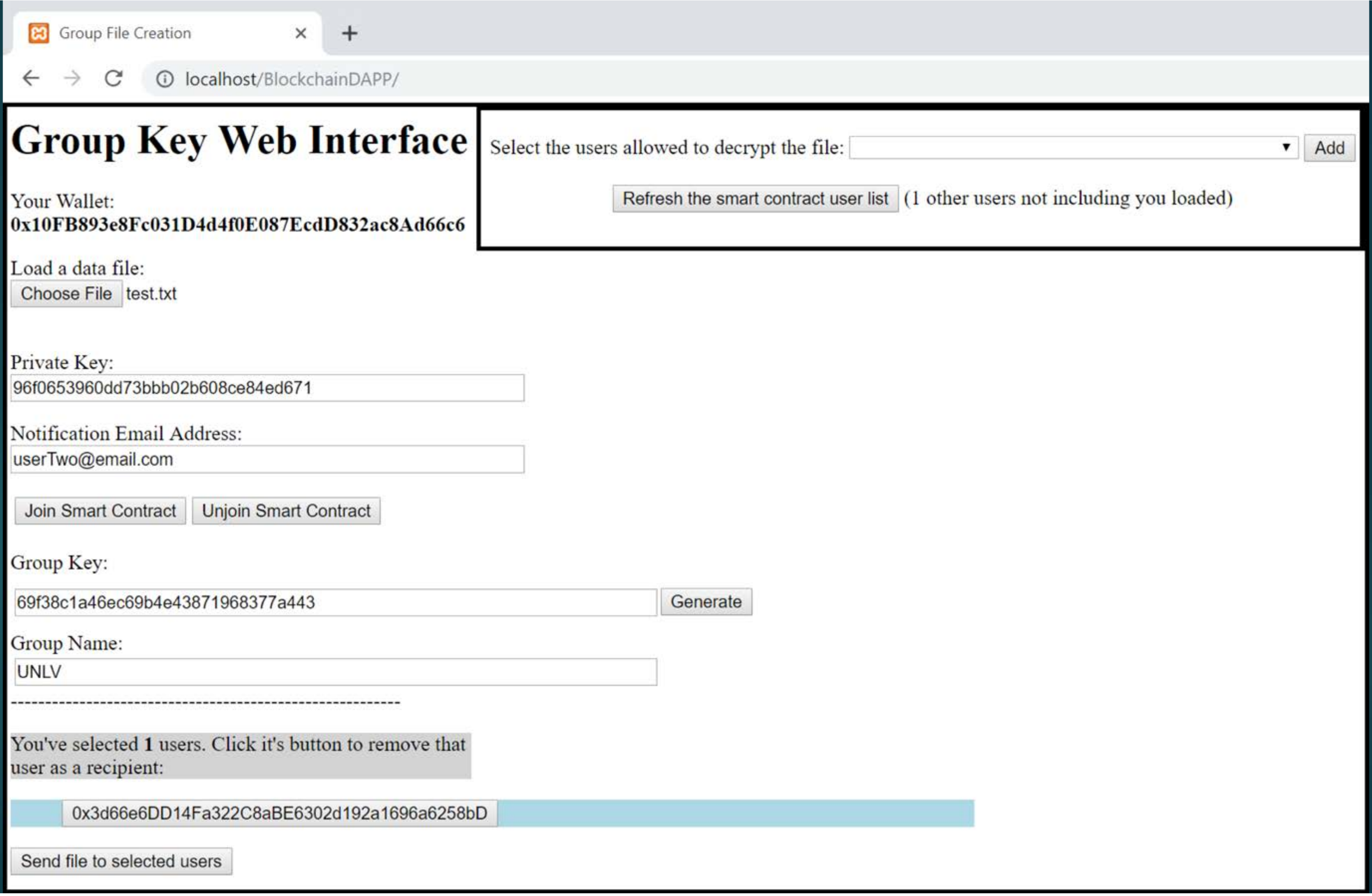
@

g

IPFS Node

IPFS Node

IPFS Node


Group File Creation
localhost/BlockchainDAPP/
Add
Refresh the smart contract user list
Choose File
Join Smart Contract
Unjoin Smart Contract
Generate
Send file to selected users

Group Key Web Interface

Select the users allowed to decrypt the file:

(1 other users not including you loaded)

Your Wallet:
0x10FB893e8Fc031D4d4f0E087EcdD832ac8Ad66c6

Load a data file:
test.txt

Private Key:
96f0653960dd73bbb02b608ce84ed671

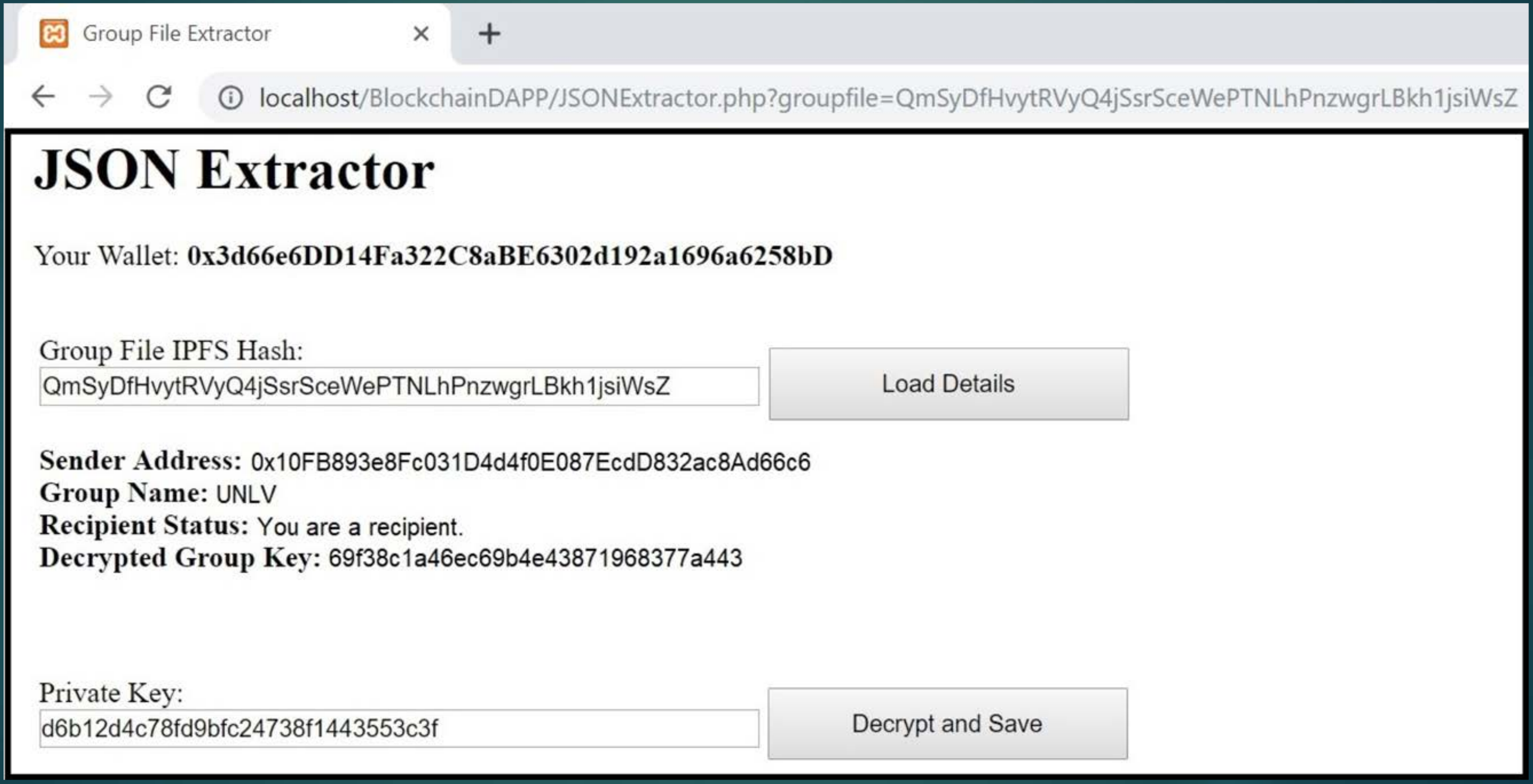
Notification Email Address:
userTwo@email.com

Group Key:
69f38c1a46ec69b4e43871968377a443

Group Name:
UNLV

You've selected **1** users. Click it's button to remove that user as a recipient:

0x3d66e6DD14Fa322C8aBE6302d192a1696a6258bD



Group File Extractor
localhost/BlockchainDAPP/JSONExtractor.php?groupfile=QmSyDfHvytRVyQ4jSsrSceWePTNLhPnzwgrLBkh1jsiWsZ

JSON Extractor

Your Wallet: **0x3d66e6DD14Fa322C8aBE6302d192a1696a6258bD**

Group File IPFS Hash:
QmSyDfHvytRVyQ4jSsrSceWePTNLhPnzwgrLBkh1jsiWsZ

Sender Address: 0x10FB893e8Fc031D4d4f0E087EcdD832ac8Ad66c6
Group Name: UNLV
Recipient Status: You are a recipient.
Decrypted Group Key: 69f38c1a46ec69b4e43871968377a443

Private Key:
d6b12d4c78fd9bfc24738f1443553c3f

Performance Analysis

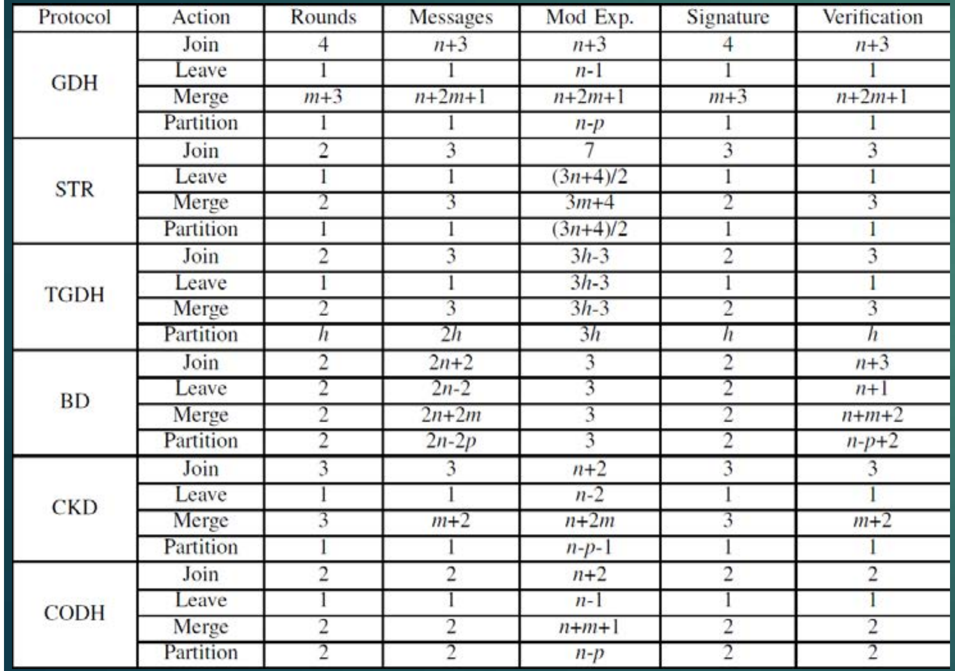
FOR GENERATING AN ENCRYPTED GROUP KEY

AND PREPARING IT FOR EMAIL

Secure Collaborative Key Management for Dynamic Groups in Mobile Networks

C. J. a. M. H. Sukin Kang
Journal of Applied Mathematics, 2014.

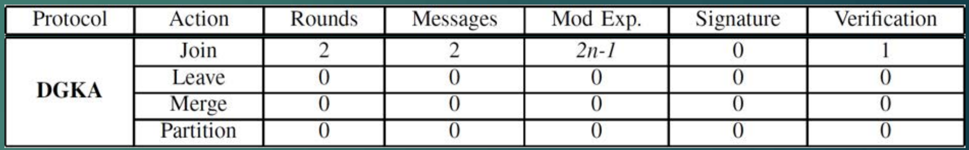
Diagram of all alternatives



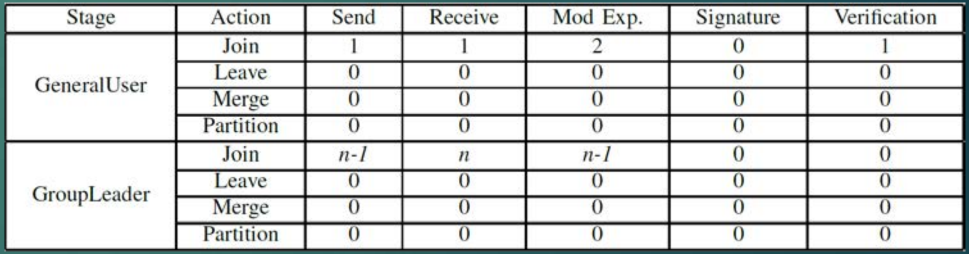
Protocol	Action	Rounds	Messages	Mod Exp.	Signature	Verification
GDH	Join	4	$n+3$	$n+3$	4	$n+3$
	Leave	1	1	n-1	1	1
	Merge	$m+3$	$n+2m+1$	$n+2m+1$	$m+3$	$n+2m+1$
	Partition	1	1	n-p	1	1
STR	Join	2	3	7	3	3
	Leave	1	1	$(3n+4)/2$	1	1
	Merge	2	3	$3m+4$	2	3
	Partition	1	1	$(3n+4)/2$	1	1
TGDH	Join	2	3	$3h$-3	2	3
	Leave	1	1	$3h$-3	1	1
	Merge	2	3	$3h$-3	2	3
	Partition	h	$2h$	$3h$	h	h
BD	Join	2	$2n+2$	3	2	$n+3$
	Leave	2	$2n$-2	3	2	$n+1$
	Merge	2	$2n+2m$	3	2	$n+m+2$
	Partition	2	$2n$-$2p$	3	2	n-$p+2$
CKD	Join	3	3	$n+2$	3	3
	Leave	1	1	n-2	1	1
	Merge	3	$m+2$	$n+2m$	3	$m+2$
	Partition	1	1	n-p-1	1	1
CODH	Join	2	2	$n+2$	2	2
	Leave	1	1	n-1	1	1
	Merge	2	2	$n+m+1$	2	2
	Partition	2	2	n-p	2	2

All of these are based on ring and binary tree designs.

Comparison with our Distributed Group Key Algorithm (DGKA)



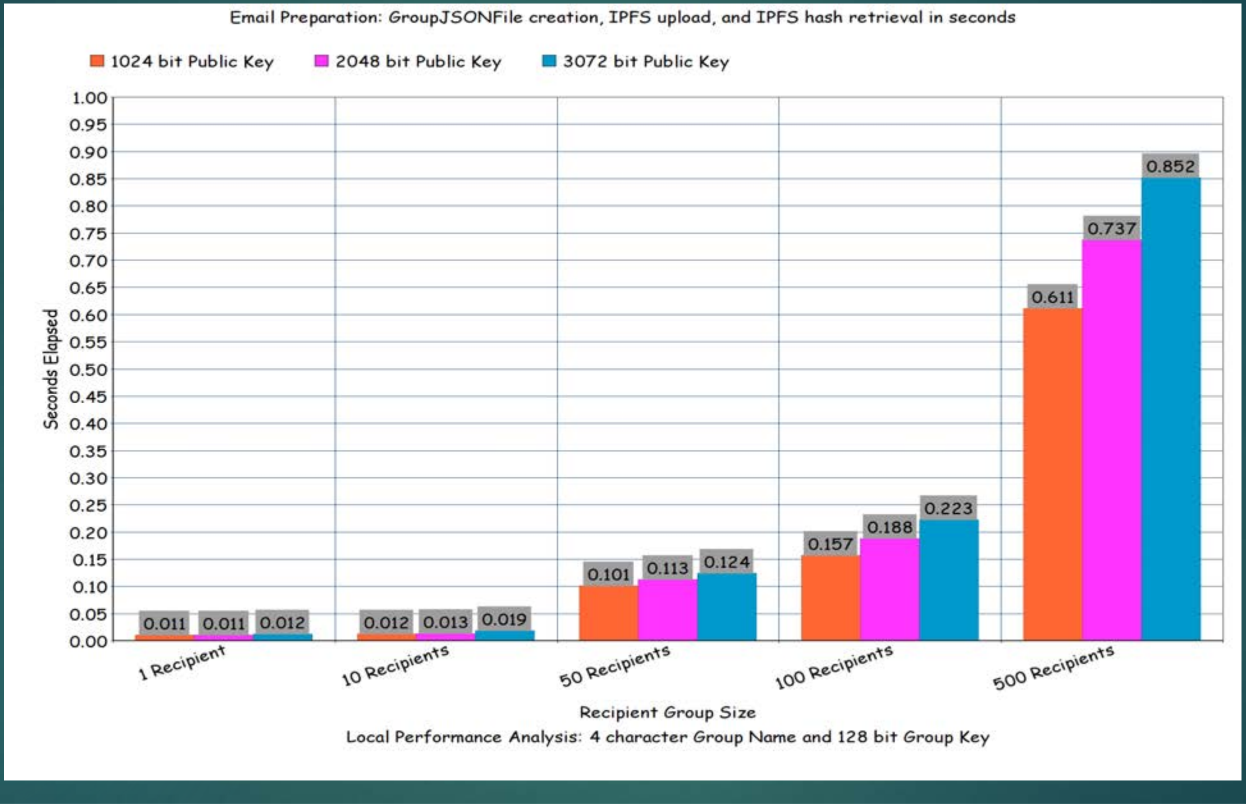
Protocol	Action	Rounds	Messages	Mod Exp.	Signature	Verification
DGKA	Join	2	2	$2n$-1	0	1
	Leave	0	0	0	0	0
	Merge	0	0	0	0	0
	Partition	0	0	0	0	0



Stage	Action	Send	Receive	Mod Exp.	Signature	Verification
GeneralUser	Join	1	1	2	0	1
	Leave	0	0	0	0	0
	Merge	0	0	0	0	0
	Partition	0	0	0	0	0
GroupLeader	Join	n-1	n	n-1	0	0
	Leave	0	0	0	0	0
	Merge	0	0	0	0	0
	Partition	0	0	0	0	0

Our protocol uses a similar amount of operations.
Our protocol performs above the standard efficiency.
Our protocol is immune to traditional vulnerabilities.

Our Results



After all of the users publish their required items to the smart contract a Group Leader may establish private group communication for **500 users** in **0.852 seconds** of time using the **3027 bit Public Key IETF standard**

Conclusion and Future Work

Conclusion

Our protocol is the only one available that operates on the open distributed blockchain.

Our protocol is more secure than the alternatives.

No software required, only a regular computer and internet access.

Future Research

Future research would enable 3 new features:

Implementing Elliptic Curve Cryptography

- A 256 bit Public Key is equivalent to a 3027 Diffie-Hellman bit Public Key meaning faster and cheaper transactions for the same security.

Implementing Perfect Forward Secrecy

- A solution for establishing long term private group communication channels that can't be decrypted if the Group Key is stolen.

Implementing dynamic Adding and Removing of users to an existing group

- A solution for private group communication channels needing to be reinstated each time a user is removed or added to an existing group.

The Grand Conclusion

Questions?