Channel Islands
CALIFORNIA STATE UNIVERSITY

Securing the Electorate: A Cryptographic Vote

Ty Danet & Brandon Artner • Professor Michael Soltys • Comp 499

Introduction

- Using Adi Shamir's cryptographic scheme based off of polynomial interpolation in a finite field as a base, we sought to create a voting system.
- Our application allows for being set up for any organizational structure. e.g.
  - A small group of top executives having complete power
  - Every single member having equal say
  - A hybrid of these two structures
- Voting organizations created via the application have the functionality to vote on and sign documents.
  - The signing of documents is done via DSA following the guidelines from FIPS 186-4.
- Organizations can be made via a interactive designer based in terminal or by a config file.

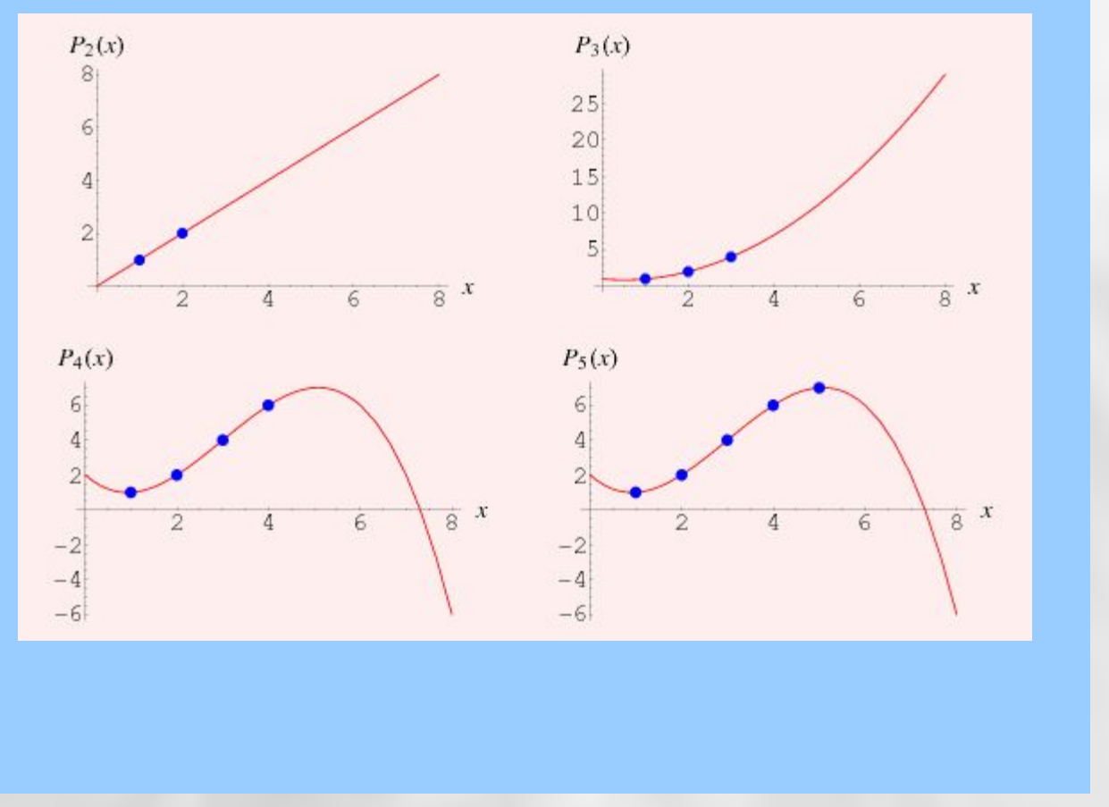
Polynomial Interpolation

$$Given\ points : (x_1, y_1), (x_2, y_2), ..., (x_k, y_k).$$

$$P(x) = \sum_{j=0}^{k} y_j l_j(x),$$

$$where\ l_j(x) = \prod_{\substack{0<i<k \\ i=j}} \frac{x - x_i}{x_j - x_i}.$$

Illustration of Polynomial Interpolation

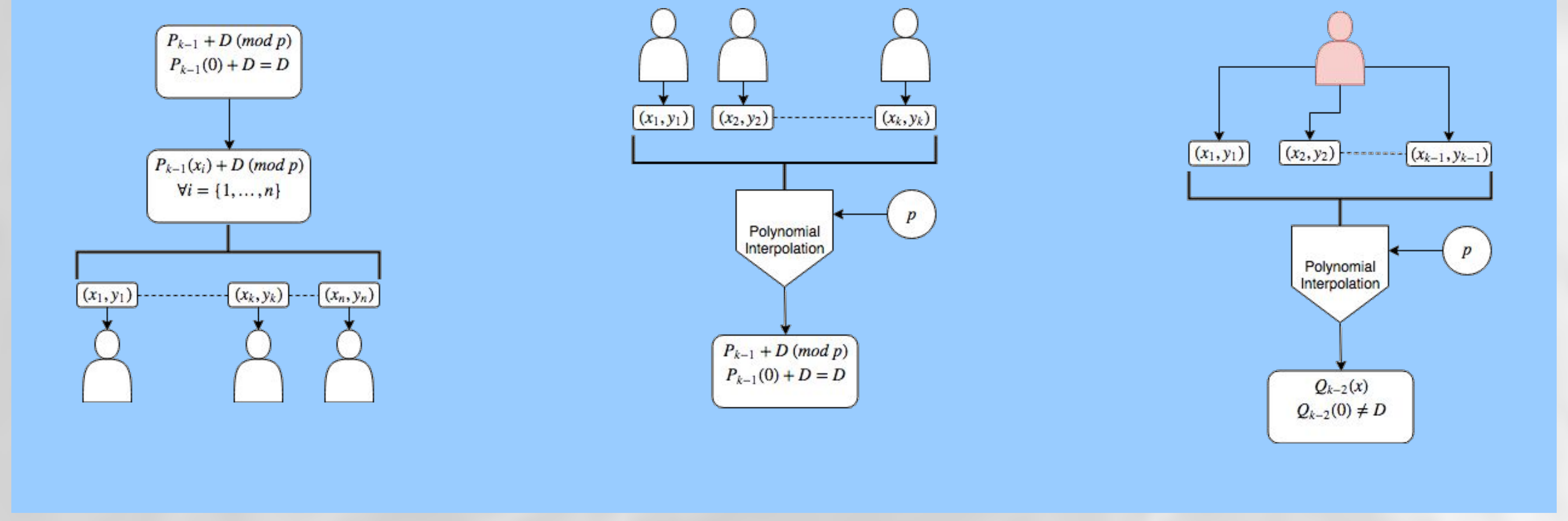


Shamir's Secret Sharing Scheme (S4)

Encryption (Key-Generation)

Decryption

Attacker

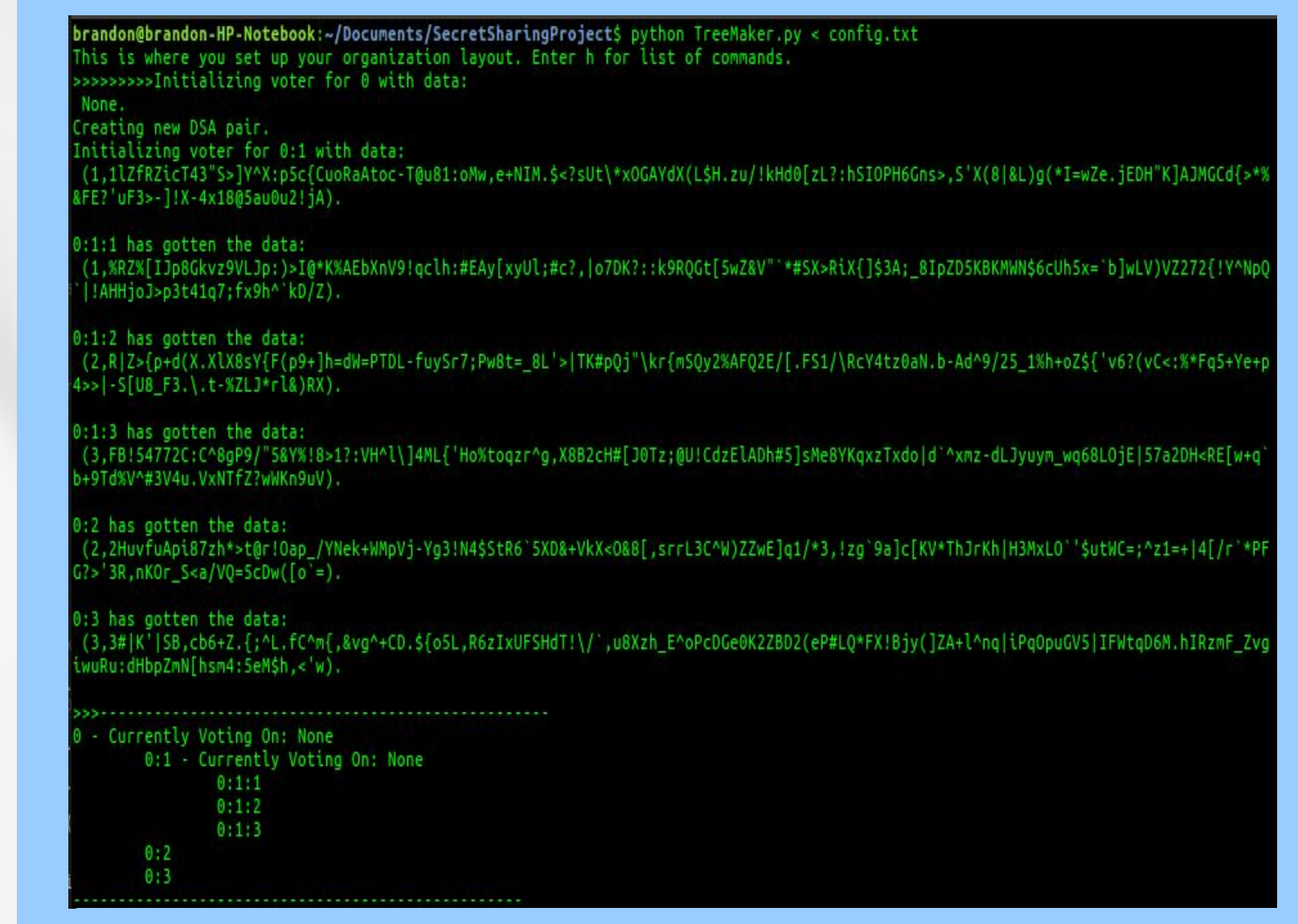


Asynchronous Neville's Method

$$\begin{array}{llllll}
x_0 & y_0 = P_0 & & & & \\
\hline
x_0 & y_0 = P_0 & & & & \\
x_1 & y_1 = P_1 & P_{0,1} & & & \\
\hline
x_0 & y_0 = P_0 & & & & \\
x_1 & y_1 = P_1 & P_{0,1} & & & \\
x_2 & y_2 = P_2 & P_{1,2} & P_{0,1,2} & & \\
\hline
x_0 & y_0 = P_0 & & & & \\
x_1 & y_1 = P_1 & P_{0,1} & & & \\
x_2 & y_2 = P_2 & P_{1,2} & P_{0,1,2} & & \\
\vdots & & & & & \\
x_k & y_k = P_k & P_{k-1,k} & & \dots & P_{0,\dots,k} = D
\end{array}$$

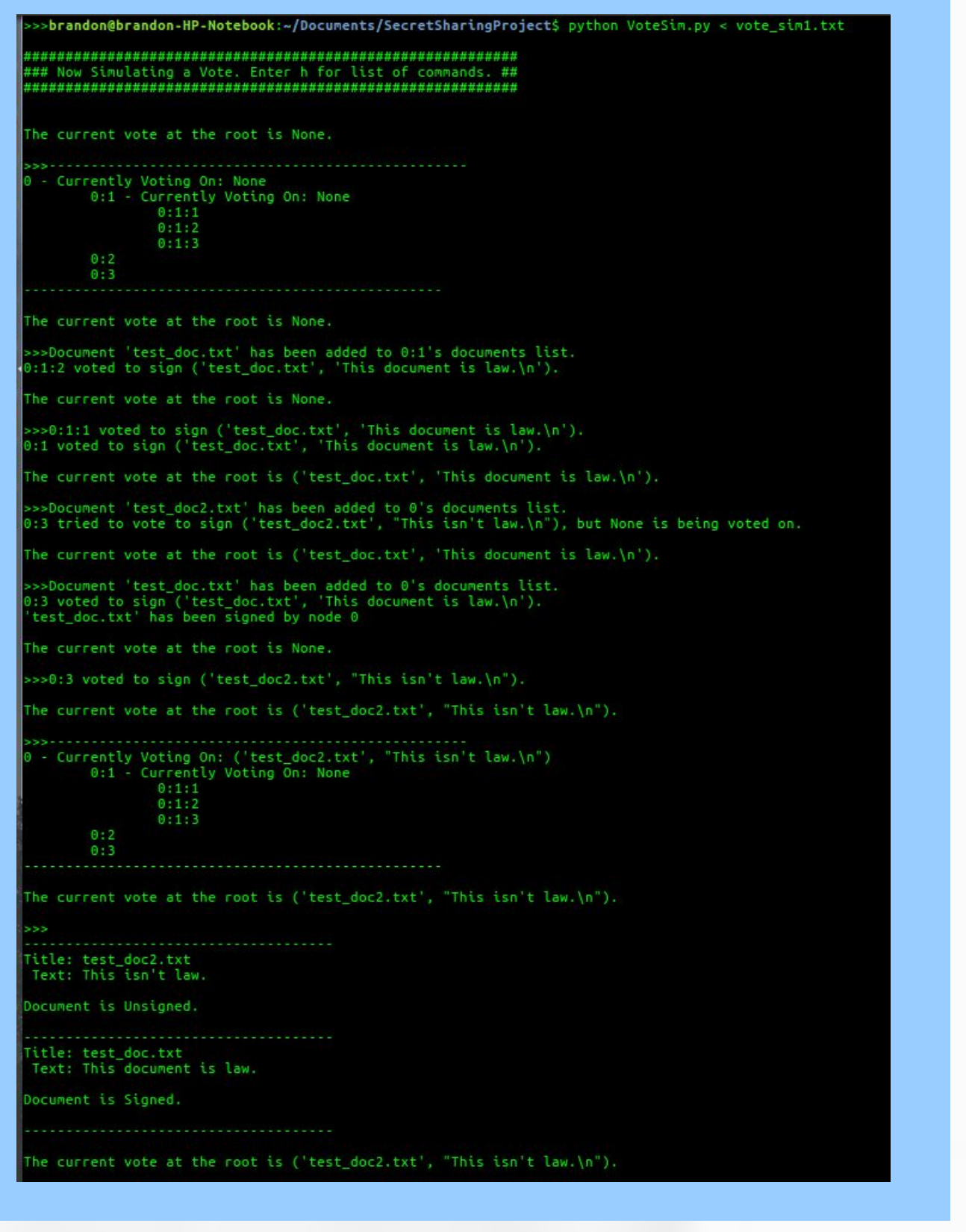
Sample Run

- Voting Application
  - Allows for interactive building of a voting organization or to use a predefined configuration
  - A Terminal application
  - Voting process which results in documents being signed if a vote passes



```
brandon@brandon-HP-Notebook:~/Documents/SecretSharingProject$ python TreeMaker.py < config.txt
This is where you set up your organization layout. Enter h for list of commands.
>>>>>>>>>Initializing voter for 0 with data:
 None.
Creating new DSA pair.
Initializing voter for 0:1 with data:
 (1,1lZfRZicT43"S>]Y^X:p5c{CuoRaAtoc-T@u81:oMw,e+NIM.$<?sUt\*xOGAYdX(L$H.zu/!kHd0[zL?:hSIOPH6Gns>,S'X(8|&L)g(*I=wZe.jEDH"K]AJMGCd{>*%
&FE?'uF3>-]!X-4x18@5au0u2!jA).

0:1:1 has gotten the data:
 (1,%RZ%[IJp8Gkvz9VLJp:)>I@*K%AEbXnV9!qclh:#EAy[xyUl;#c?,|o7DK?::k9RQGt[5wZ&V"`*#SX>RiX{]$3A;_8IpZD5KBKMWN$6cUh5x=`b]wLV)VZ272{!Y^NpQ
`|!AHHjoJ>p3t41q7;fx9h^`kD/Z).

0:1:2 has gotten the data:
 (2,R|Z>{p+d(X.XlX8sY{F(p9+]h=dW=PTDL-fuySr7;Pw8t=_8L'>|TK#pQj"\kr{mSQy2%AFQ2E/[.FS1/\RcY4tz0aN.b-Ad^9/25_1%h+oZ${'v6?(vC<:%*Fq5+Ye+p
4>>|-S[U8_F3.\.t-%ZLJ*rl&)RX).

0:1:3 has gotten the data:
 (3,FB!54772C:C^8gP9/"5&Y%!8>1?:VH^l\]4ML{'Ho%toqzr^g,X8B2cH#[J0Tz;@U!CdzElADh#5]sMe8YKqxzTxdo|d`^xmz-dLJyuym_wq68LOjE|57a2DH<RE[w+q`
b+9Td%V^#3V4u.VxNTfZ?wWKn9uV).

0:2 has gotten the data:
 (2,2HuvfuApi87zh*>t@r!0ap_/YNek+WMpVj-Yg3!N4$StR6`5XD&+VkX<0&8[,srrL3C^W)ZZwE]q1/*3,!zg`9a]c[KV*ThJrKh|H3MxLO`'$utWC=;^z1=+|4[/r`*PF
G?>'3R,nKOr_S<a/VQ=5cDw([o`=).

0:3 has gotten the data:
 (3,3#|K'|SB,cb6+Z.{;^L.fC^m{,&vg^+CD.${o5L,R6zIxUFSHdT!\/`,u8Xzh_E^oPcDGe0K2ZBD2(eP#LQ*FX!Bjy(]ZA+l^nq|iPqOpuGV5|IFWtqD6M.hIRzmF_Zvg
iwuRu:dHbpZmN[hsm4:5eM$h,<'w).

>>>-------------------------------------------------
0 - Currently Voting On: None
        0:1 - Currently Voting On: None
                0:1:1
                0:1:2
                0:1:3
        0:2
        0:3
-------------------------------------------------
```

These images are sample runs of the voting system without the gui. Left: A sample of the interactive designing of a voting organization. Right: A sample of a voting session, where one document is signed and another vote is initiated.



```
>>>brandon@brandon-HP-Notebook:~/Documents/SecretSharingProject$ python VoteSim.py < vote_sim1.txt

############################################################
### Now Simulating a Vote. Enter h for list of commands. ##
############################################################


The current vote at the root is None.

>>>-----------------------------------------------------
0 - Currently Voting On: None
        0:1 - Currently Voting On: None
                0:1:1
                0:1:2
                0:1:3
        0:2
        0:3
-----------------------------------------------------

The current vote at the root is None.

>>>Document 'test_doc.txt' has been added to 0:1's documents list.
0:1:2 voted to sign ('test_doc.txt', 'This document is law.\n').

The current vote at the root is None.

>>>0:1:1 voted to sign ('test_doc.txt', 'This document is law.\n').
0:1 voted to sign ('test_doc.txt', 'This document is law.\n').

The current vote at the root is ('test_doc.txt', 'This document is law.\n').

>>>Document 'test_doc2.txt' has been added to 0's documents list.
0:3 tried to vote to sign ('test_doc2.txt', "This isn't law.\n"), but None is being voted on.

The current vote at the root is ('test_doc.txt', 'This document is law.\n').

>>>Document 'test_doc.txt' has been added to 0's documents list.
0:3 voted to sign ('test_doc.txt', 'This document is law.\n').
'test_doc.txt' has been signed by node 0

The current vote at the root is None.

>>>0:3 voted to sign ('test_doc2.txt', "This isn't law.\n").

The current vote at the root is ('test_doc2.txt', "This isn't law.\n").

>>>-----------------------------------------------------
0 - Currently Voting On: ('test_doc2.txt', "This isn't law.\n")
        0:1 - Currently Voting On: None
                0:1:1
                0:1:2
                0:1:3
        0:2
        0:3
-----------------------------------------------------

The current vote at the root is ('test_doc2.txt', "This isn't law.\n").

>>>
-------------------------------------
Title: test_doc2.txt
 Text: This isn't law.

Document is Unsigned.

-------------------------------------
Title: test_doc.txt
 Text: This document is law.

Document is Signed.

-------------------------------------

The current vote at the root is ('test_doc2.txt', "This isn't law.\n").
```

Level of Security

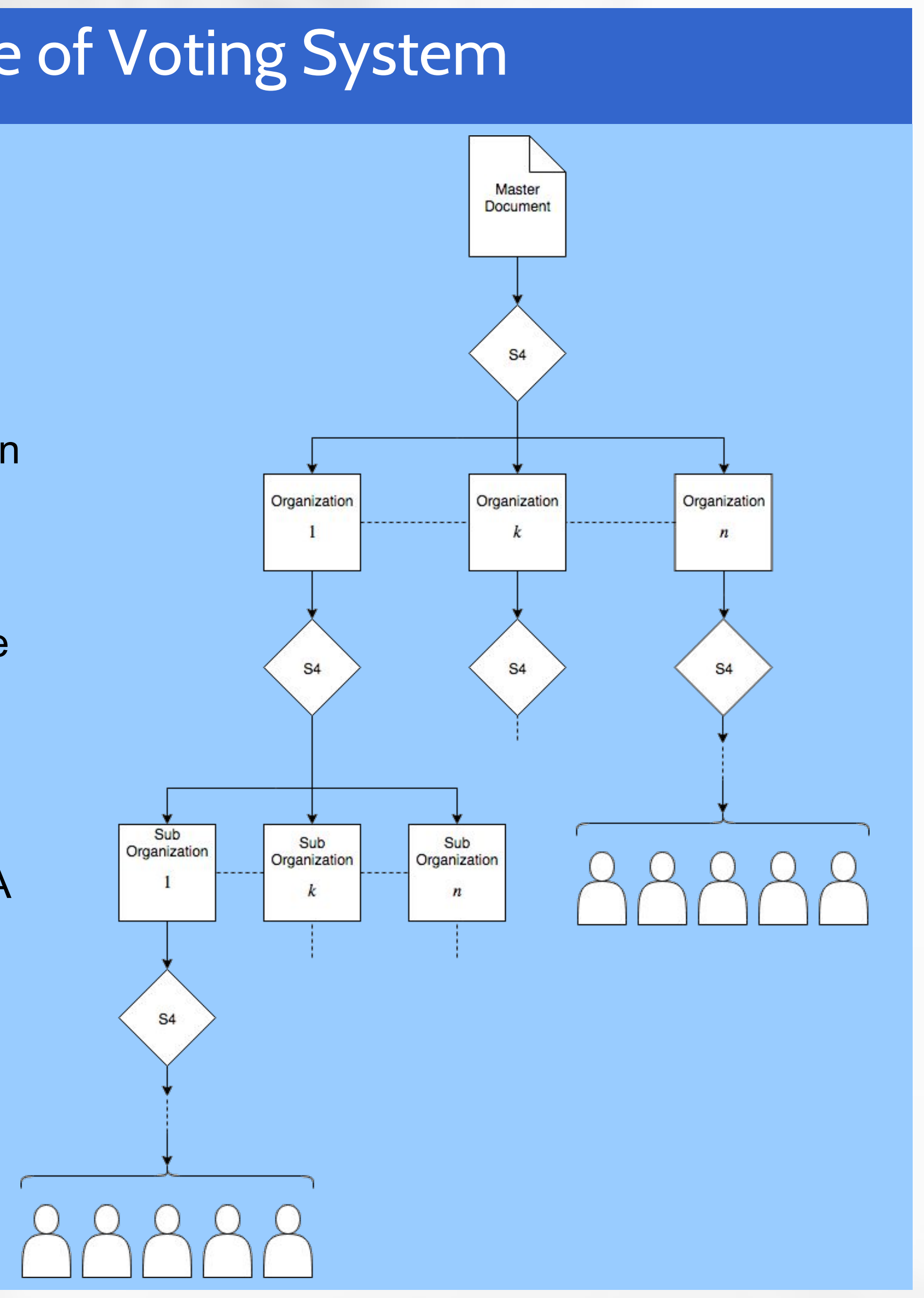
- S4 is both Information-Theoretic Secure and Perfectly Secure.
- Information-Theoretic Secure
  - Even with infinite computing power S4 could not be broken.
  - However, since we use the secret data as a private key for DSA signatures, our system loses this quality, but it is still as secure as DSA which is secure.
- Perfectly Secure
  - If there is ciphertext produced that uses it, no information about the plaintext is provided without knowledge of the key.

Structure of Voting System

Shamir's Scheme can be extended into a hierarchical structure.
We can simply compute
f(x, y) = D_i (mod p),
which translates a key-pair into an integer, to use as the secret data for the lower tier of the hierarchy.

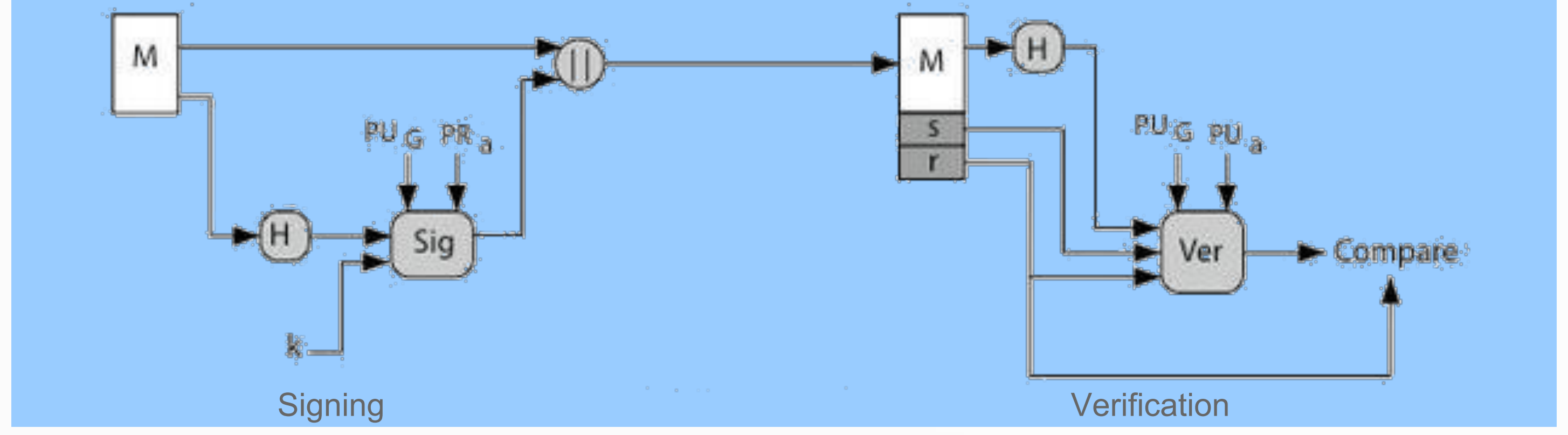
Then, when voting, to recover the data a reverse function is used,
g(D_i) = (x, y).

The digital signing for the documents sets is done with DSA as described by DSS.



Digital Signature Algorithm (DSA)

- Given primes *p,q;* generator *g;* and private-public key pair (*x,y*).
- Select a hashing function *H.* (We chose SHA-2)
- Signing a message m:
  - Randomly generate a value *k* where 1<*k*<*q*.
  - Calculate *r* = (g^k *(mod p)*) *(mod q).* (Ensure *r* ≠ 0)
  - Calculate *s* = k^{-1}*(H(m) + xr) (mod q).* (Ensure *s* ≠ 0)
  - Return the signature: *(r, s).*
- Verifying the signature for the message m:
  - Calculate:
    - *w = s^{-1} (mod q).*
    - *u_1 = H(m)×w (mod q).*
    - *u_2 = r×w (mod q).*
    - *v = (g^{u1}×y^{u2} (mod p)) (mod q).*
  - If *v = r,* verified; else, not verified.



Attack Vectors and Implications

- False keys
  - Duplicate Keys: Solved
  - Invalid Keys: Could be solved by assigning public-key pairs
- Theoretical 100% voter turnout
  - Since a vote passes only with enough votes, abstention is equivalent to voting no
  - This makes it difficult for an active minority to pass laws taking advantage of low voter turnout
- Any single voter located higher in the voting structure has more power than any single voter located lower in the structure.
  - Simply make all individual voters exist on the same level

Future Work

- Move from simulations on a single machine to simulations using networking
- Further analysis of how secure the cryptography is?
- Convert into a Web App?
- Have each level inherit the documents from above, and able to make additions that lower levels will inherit, like federal/state laws

Literature Cited

- Shamir, Adi. "How to Share a Secret." Communications of the ACM, vol. 22, no. 11, Jan. 1979, pp. 612–613., doi:10.1145/359168.359176.
- Blakley, G.R. Safeguarding cryptographic keys. Proc. AFIPS 1979 NCC, Vol. 48, Arlington, Va., June 1979, pp.
  313-317.
- Burden, Richard L., et al. Numerical Analysis. 10th ed., Cengage Learning, 2016.
- FIPS 186-4

Acknowledgements

- Dr. Soltys - advisor
- Maria - Scrum Master
- Adi Shamir - The S of RSA