Pseudonym and Anonymous Credential Systems

Kyle Soska – 4/13/2016

Moving Past Encryption

- Encryption Does:
  - Hide the contents of messages that are being communicated
  - Provide tools for authenticating messages

- Encryption Does Not:
  - Hide who is communicating with who
  - Hide an upper bound on how much they are communicating
  - Hide timing information or other aspects of the communication

Sensitive Information

(Name, Birthdate, SSN, Patient File, …)

(Name, Birthday, SSN, Employee Review)

Employer

Alice

(Name, Birthdate, SSN, Prescriptions, …)

(Name, Birthdate, SSN, DNA Fingerprint, …)

Information Sharing Concerns

SELECT * FROM patients WHERE
Name = 'Alice' and SSN = 123-45-6789

(Alice, 1-1-1970, 123-45-6789, Patient File, …)

- An employer and a hospital could share information to give the employer Alice's medical records
- The employer could learn that Alice is going to have a baby soon or that she has some illness and choose to fire her

Sensitive Information

- Problem: Alice uses her real identity (personally identifying information) to authenticate to different organizations
- These organizations can collude and share data to learn a lot about Alice that she does not want them to know
  - Employer learns that she is going to have a baby
  - Insurance company learns that she has a genetic pre-disposition for cancer
  - Etc.
- Question: How do we resolve this problem?
- Idea: Don't use real personal information to authenticate to these organizations

Sensitive Information

(xXA1ic3Xx, Birthdate, Patient File, …)

(xXA1ic3Xx, Birthday, Employee Review)

Employer

xXA1ic3Xx

(xXA1ic3Xx, Birthdate, Prescriptions, …)

(xXA1ic3Xx, Birthdate, DNA Fingerprint, …)

Sensitive Information

- **Problem:** Even if Alice uses a Nym not connected with her real identity, if she uses the same Nym with different organizations, then data-sharing attacks are still possible

- Data sharing attacks are leverage the fact that Alice's Nyms are linkable, information associated with one of her Nyms can be linked to her other Nyms

- Idea: Use a different Nym for each organization

Nym Linkability

ID	DOB	SSN
Alice	1-1-1970	123-45-6789
Bob	1-1-2000	111-11-1111
Charlie	12-31-1999	555-55-5555
David	7-7-1970	777-77-777

Different name but same value for a unique field, and same birthday

ID	DOB	SSN
April	4-20-1996	001-01-1101
Alyssa	1-1-1970	123-45-6789
Charlie	12-31-1999	555-55-5555
Don	8-8-1991	999-99-9999

Sensitive Information

(aLiCe, Birthdate, Patient File, …)

(xXA1ic3Xx, Birthday, Employee Review)

(yasuo_only, Birthdate, Prescriptions, …)

{xXA1ic3Xx, aLiCe, yasuo_only, rengar_only}

Employer

(rengar_only, Birthdate, DNA Fingerprint, …)

Sensitive Information

- Problem: What happens when different organizations do need to communicate?
  - Ex. Hospital needs to transfer prescriptions to pharmacy
  - We want selective information disclosure

- Problem: Users can share identities with each other
  - Alice wants to share her medical insurance with all of her friends

How to share information?

(aLiCe, Birthdate, Patient File, …)

(yasuo_only, Birthdate, Prescriptions, …)

Prescription for aLiCe

{aLiCe, yasuo_only}

I don't know who aLiCe is, you are yasuo_only

If prescriptions written for aLiCe were able to be redeemed by yasuo_only, then Alice could sell her prescription to someone else, or her prescription could be stolen etc.

Paradox of Information Sharing & Unlikability

- Organization 1 and Organization 2 want to exchange important information about Alice
  - Ex. A Drug Prescription

- The organizations need to make sure they are referring to the same person, (linkable)
  - The pharmacy needs to make sure that Alice is really the person that the prescription was written for

- Alice's identities need to be unlinkable so that nothing but the allowed information can be shared

Crypto Magic To The Rescue

- Alice will generate a single master key (public, private)
- Alice will register her keypair with a trusted CA, her keypair will be her nym with the CA
- Alice establishes a different nym with each organization such that her interactions with each organization are unlinkable
  - Does not consider timing information or side channels
- An organization can grant Alice a **credential** that attests to some property
- Alice can convince another organization of some property by showing them a credential that was previously granted to Alice
  - This process is referred to as transferring a credential

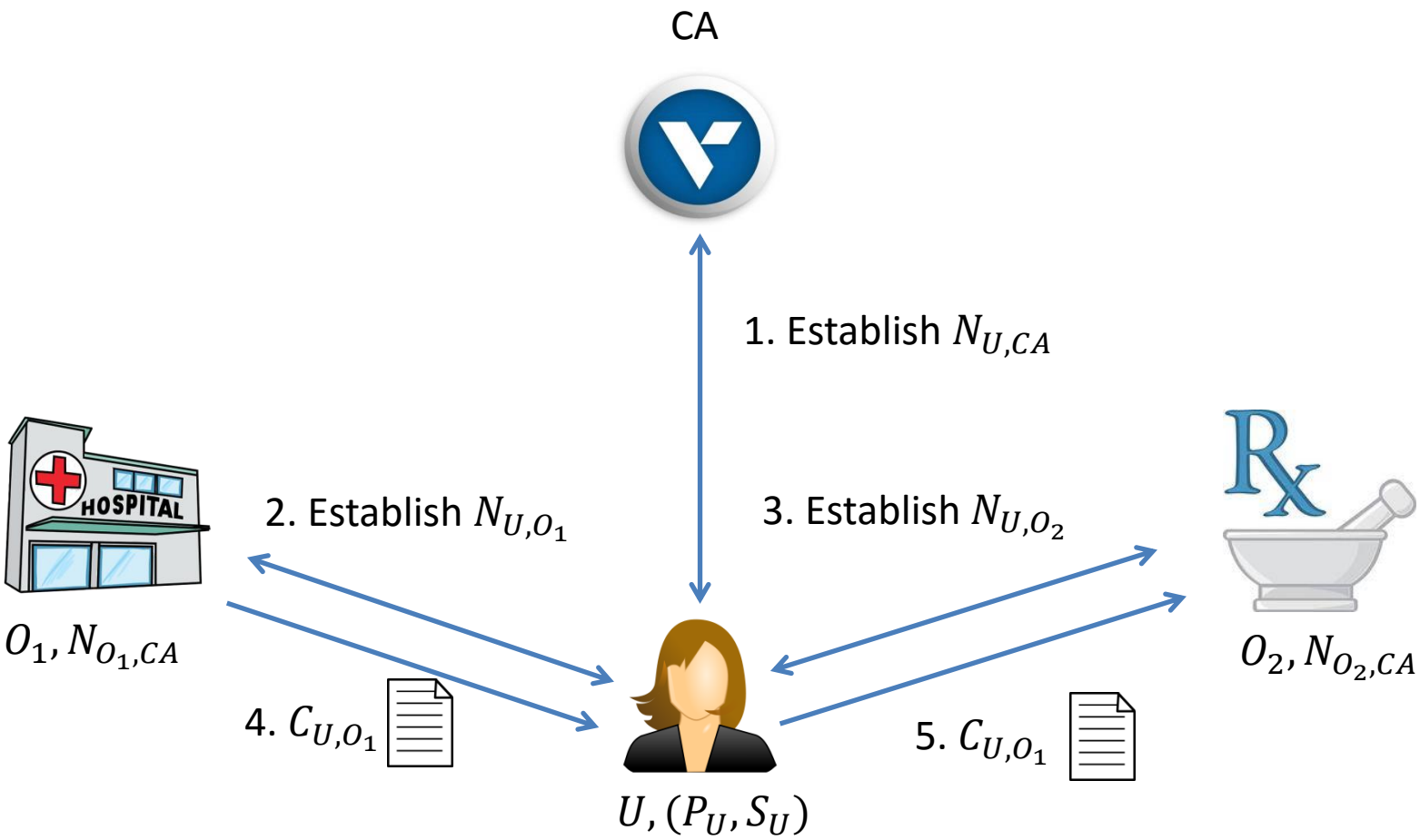
Actors and Objects

- $\boldsymbol{CA}$: Unique certification authority, trusted by all actors in the system
- $\boldsymbol{U}$: A user (Possibly many users)
  - $\boldsymbol{P_U, S_U}$: Master public key and secret key of U
  - $N(U, O)$: Set of nyms U has generated with O
  - $N(U)$: Set of nyms U has generated with anyone

- $\boldsymbol{O}$: An organization (Possibly many organizations)
  - $\boldsymbol{P_O, S_O}$: Master public key and secret key of O
  - $\boldsymbol{P_O^C, S_O^C}$: Public and secret key of O for credential C
  - $\boldsymbol{N(O)}$**:** Set of nyms O has generated with any user

- $N_{U,O}$: User U's nym with organization O
- Gen_U: Asymmetric key generation algorithm for generating master keypair

System Overview



Intuitive Statement of Goals

1. We want a system where users can create pseudonyms with different organizations, possibly multiple pseudonyms with the same organization

2. No set of organizations can collaborate to link pseudonyms of a user, an organization cannot link the multiple pseudonyms from the same user

3. A user can prove a statement from one organization to another organization using credential transfer
    1. Ex. The hospital has granted a prescription for Alice to the pharmacy

4. No set of users or organizations can forge a credential

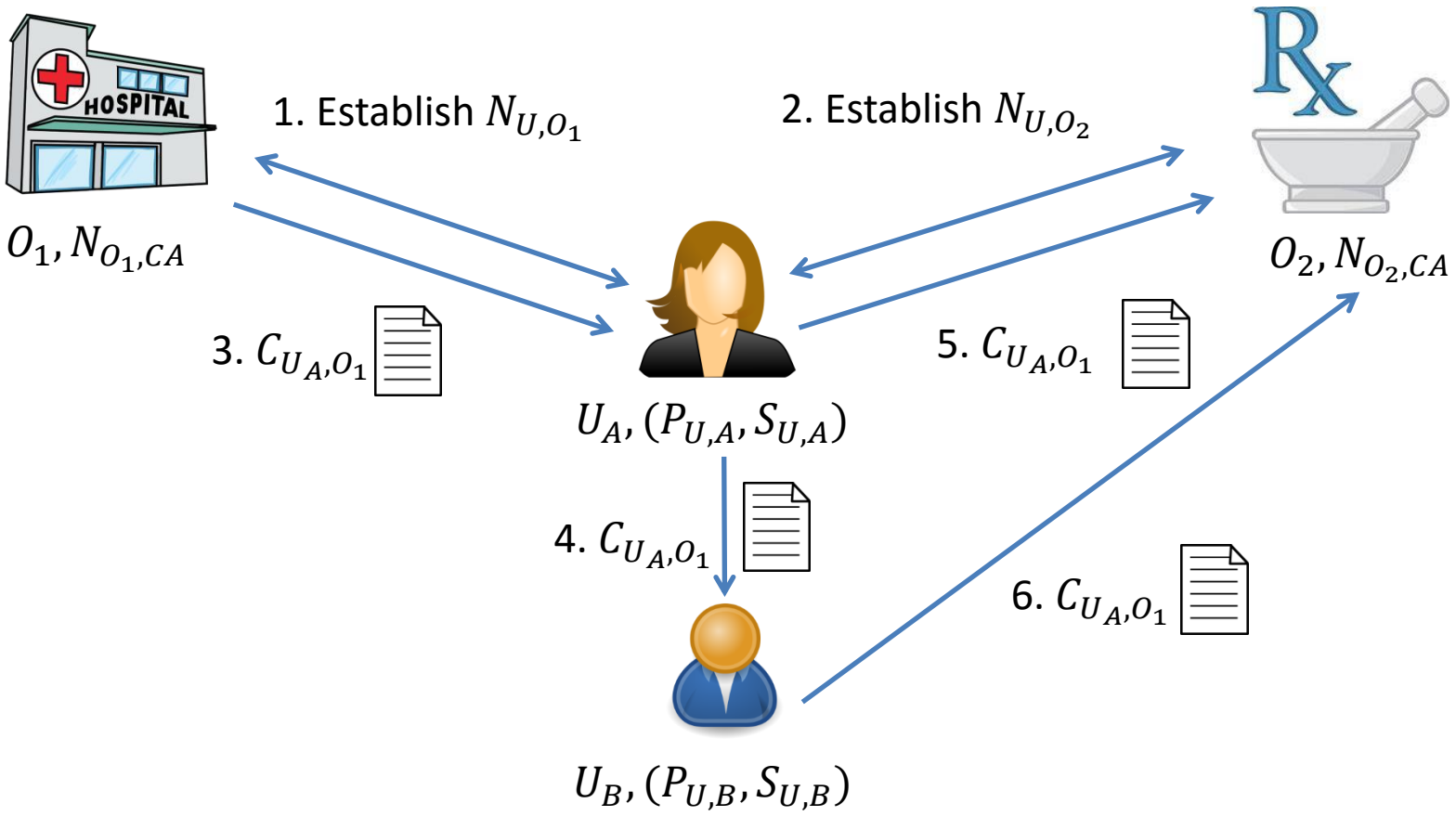
5. Users cannot share credentials with each other
    1. A user cannot give their health insurance to a friend

User Master Key Generation

- **User master key generation:** The user generates a master keypair derived from the discrete log problem
  - $p = 2q + 1$ for p, q large k-bit prime numbers
  - $G_q = \left|QR(\mathbb{Z}_p)\right| = q$ is the quadratic residue subgroup of $\mathbb{Z}_p$ which has order q
  - Let $g \in G_q$ be a public generator
  - User selects $x \leftarrow_R \mathbb{Z}_q$ and computes $g^x \ mod \ p$
  - User's Private Key: x
  - User's Public Key: $g^x \ mod \ p$

- The user would share this public key with the CA. The CA checks that Alice is a real person and that she has not already registered an account with the system

Transfer of Credentials Problem



- We desire that Alice can 'redeem' C_{U_A,O_1}, but not Bob
- How can we achieve this? What is the difference between Alice and Bob?

Transfer of Credentials Problem

- U_A and U_B have different nyms at O_1 and O_2, namely $N_{U_A,O_1} \neq N_{U_B,O_2}$, $N_{U_A,O_2} \neq N_{U_B,O_2}$
  - What if the credential C_{U_A,O_1} carries information about N_{U_A,O_1}?
  - What if the credential C_{U_A,O_1} carries information about N_{U_A,O_2}?
  - Credentials are supposed to be unlinkable, so tying the credential to the user's nyms seems hopeless
- $(P_{U,A}, S_{U,A}) \neq (P_{U,B}, S_{U,B})$
  - What if the credential C_{U_A,O_1} carries information about $S_{U,A}$?
  - What if the credential C_{U_A,O_1} carries information about $P_{U,A}$?
  - Secret keys must be kept secret and public keys can be forged by anyone since they are public, seem stuck

User Master Key

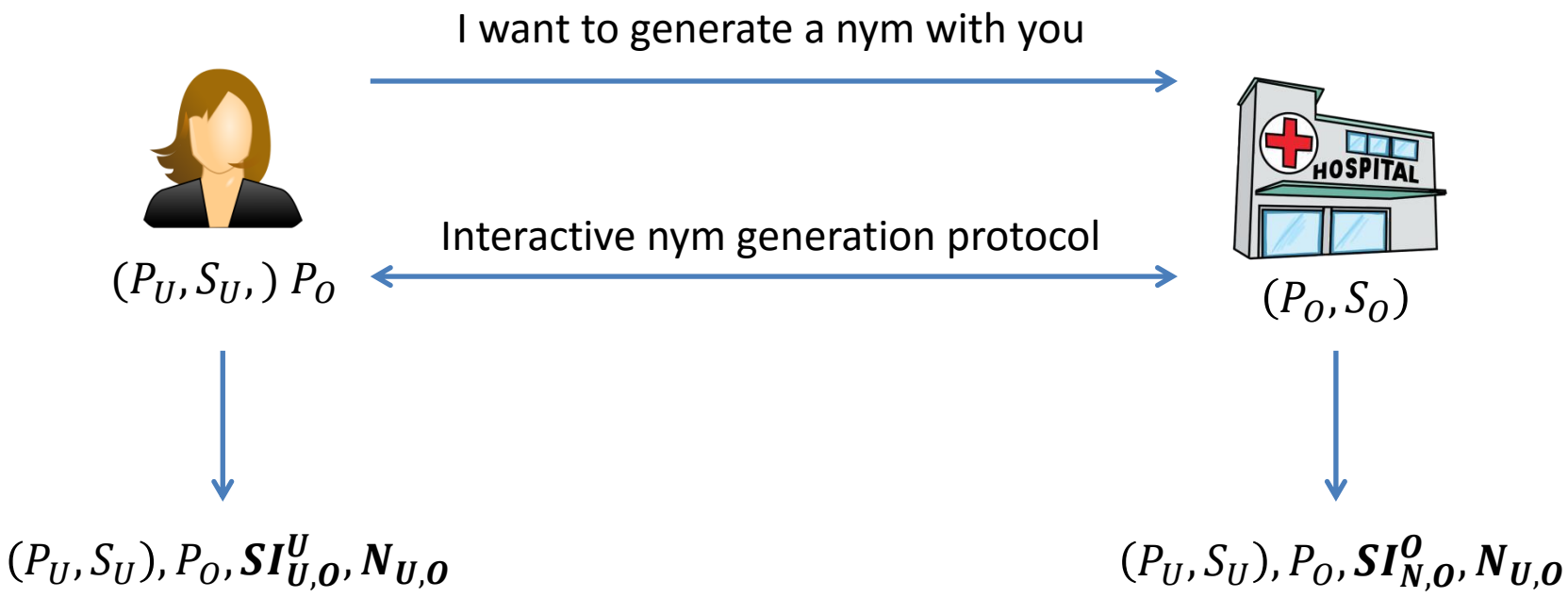
- All of the actions that a user performs are somehow tied to their master secret key
- A users nym with the CA is their public key
- A user's nyms with other organizations are derived from their master secret key
- Transferring a credential requires computations with the master secret key
  - Corollary: sharing a credential requires sharing the master secret key which is sufficient for identity theft

Nym Generation

- Secure interactive protocol between two parties U: (P_U, S_U), O: (P_O, S_O)

- Public Input: P_O, the public key of the organization
- User's Private Input: (P_U, S_U)
- Organization's Private Input: S_O
- Common Output: $N_{U,O}$
- Private User Output: $SI_{U,O}^{U}$
- Private Organization Output: $SI_{N,O}^{O}$

Nym Generation



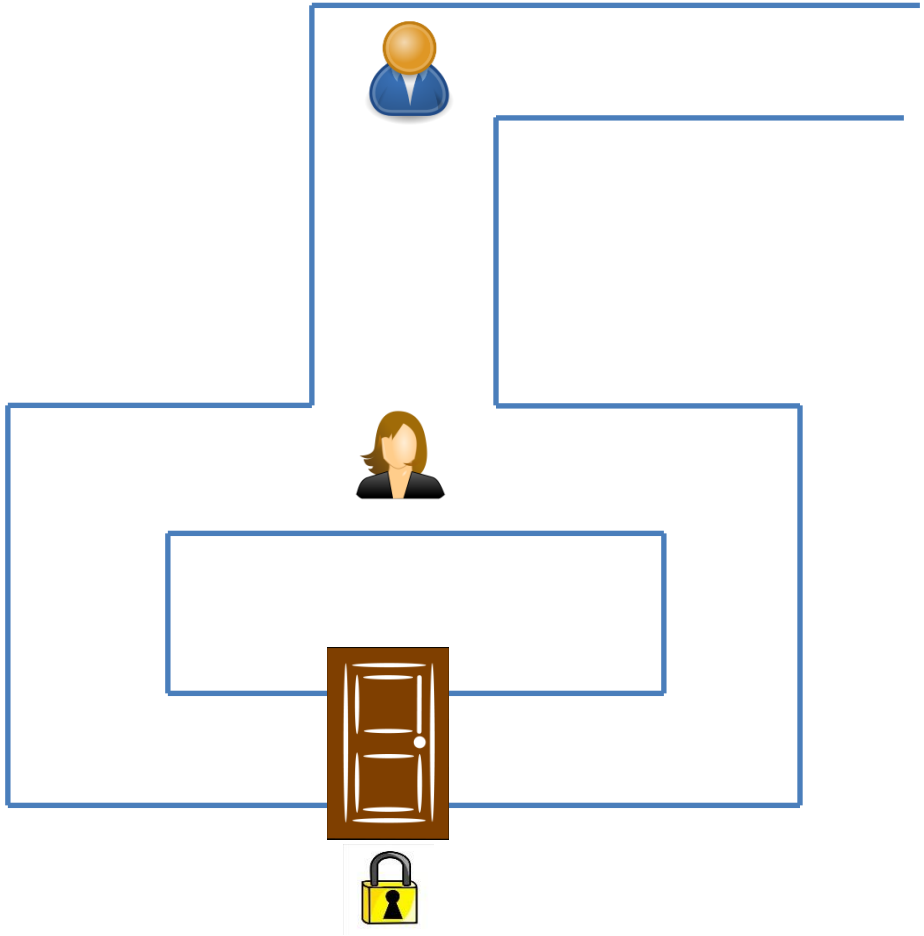
Zero Knowledge Proofs

- Interactive protocol between a prover P and a verifier V
- P wants to prove to V that he knows something, but does not want to reveal what that something is

- **Soundness**: P cannot prove false statements to the V
- **Completeness**: Proofs of true statements by P will be accepted by V

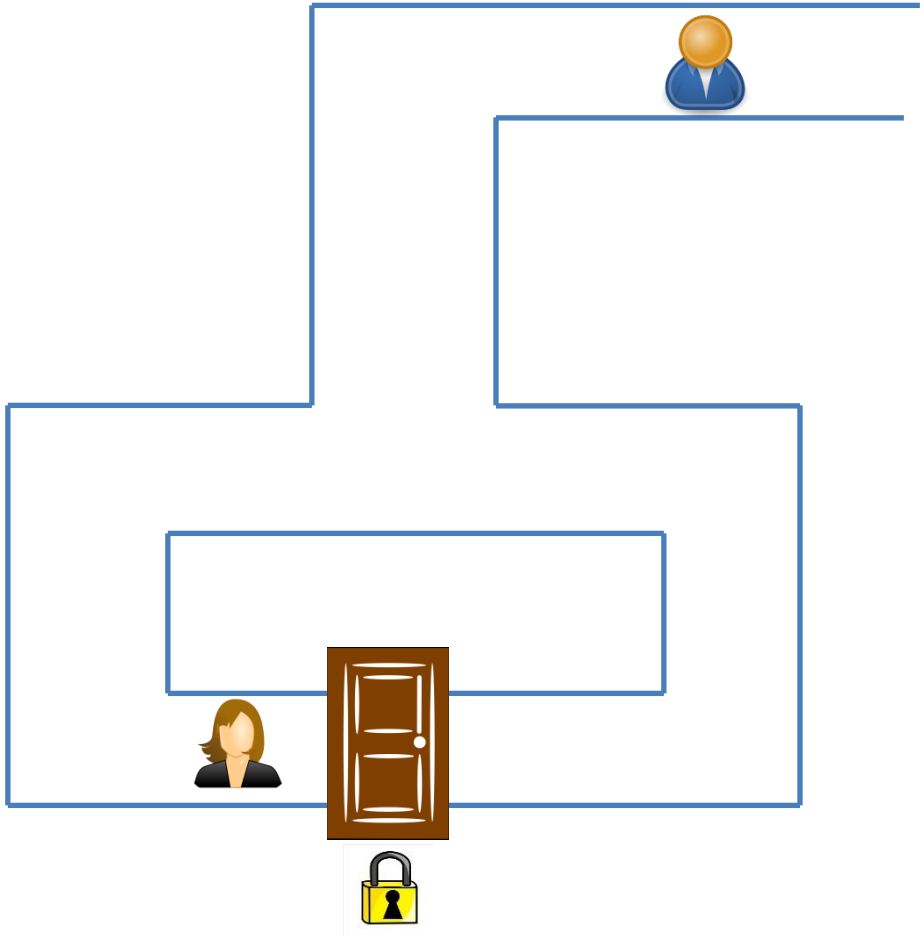
- **Zero Knowledge**: V will not learn anything other than the truth of the statement being proven

ZK Proof Example



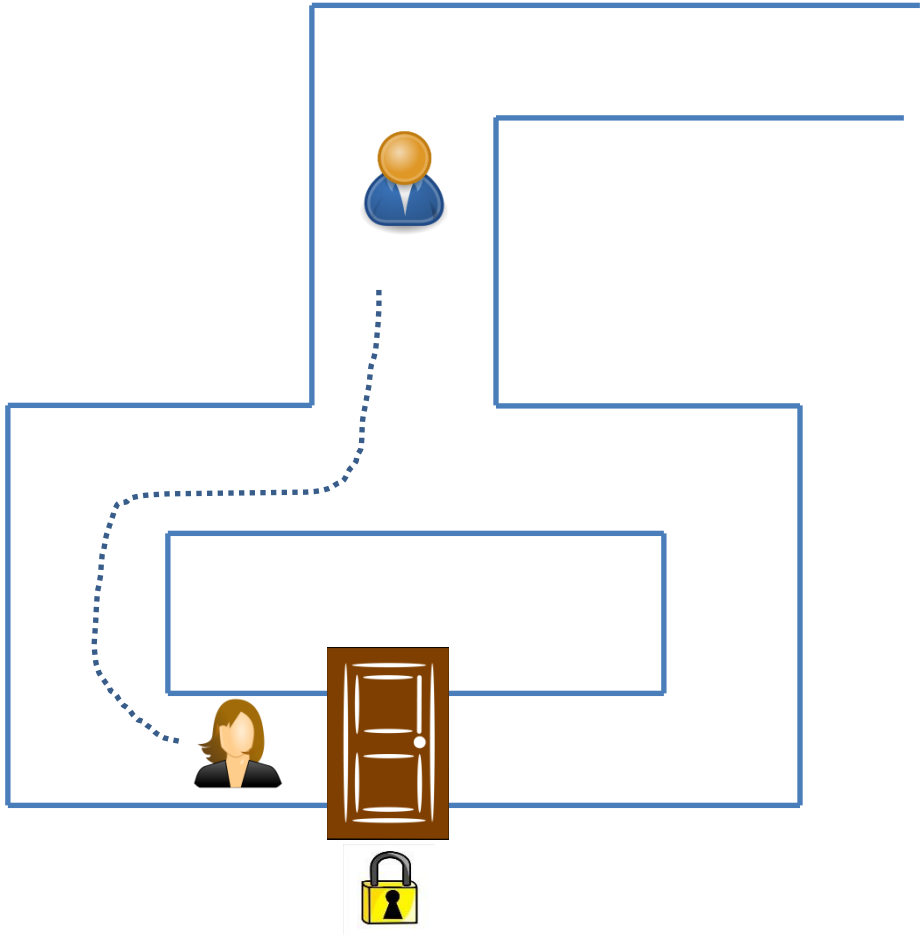
- Alice (the prover P) wants to prove to Bob (the verifier V) that she knows how to unlock the door
- If she let him watch her open the door, it would convince him that she knows how, but he might learn something about how she does it
- Instead they devise the following game to convince Bob that Alice knows how to unlock the door
- Start with a locked door

ZK Proof Example



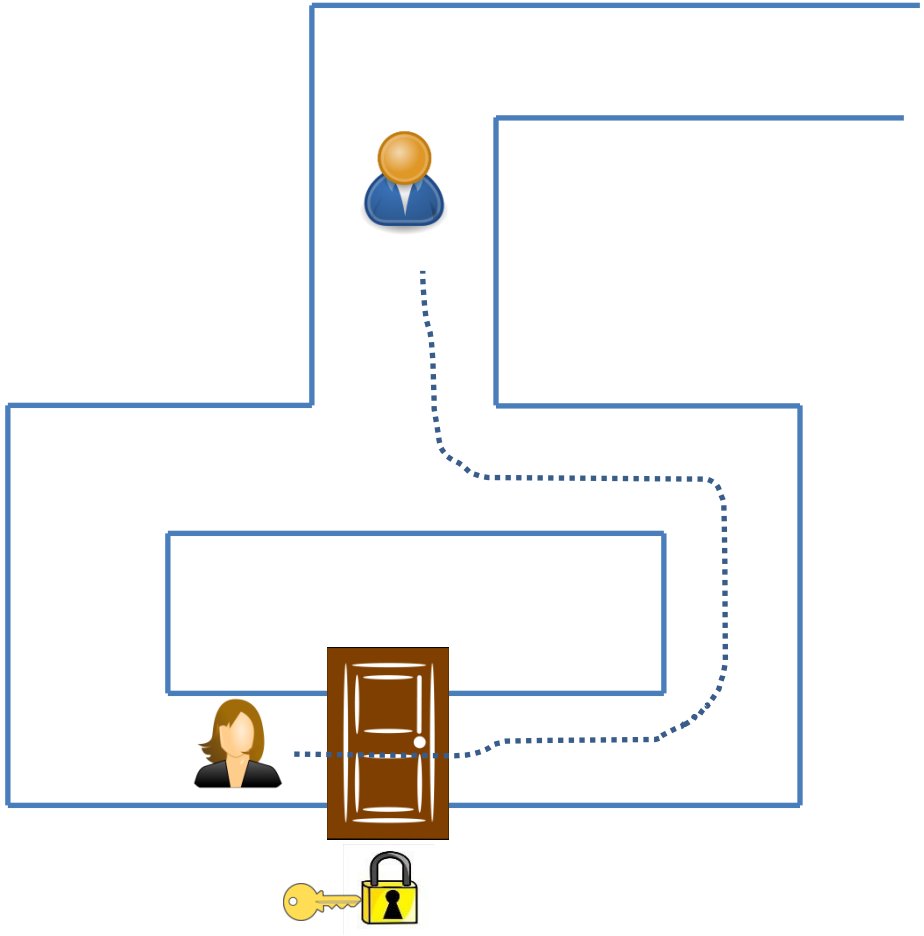
- Bob goes and hides and lets Alice pick one of the hallways to walk down
- Alice flips a coin and picks either left or right to walk down
  - Heads = Left
  - Tails = Right

ZK Proof Example



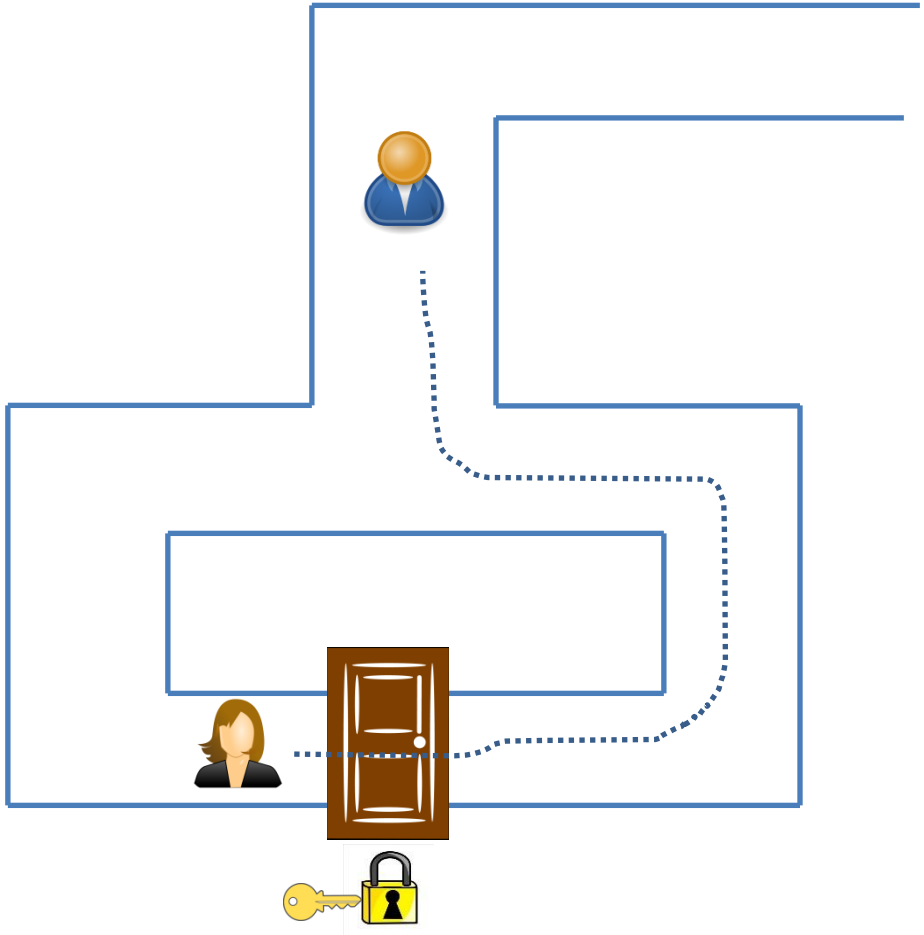
- Bob flips a coin
  - Heads = Left
  - Tails = Right
- Bob then yells down the hallway and demands that Alice appear from that side
- If Alice is already on the same side she simply walks out

ZK Proof Example



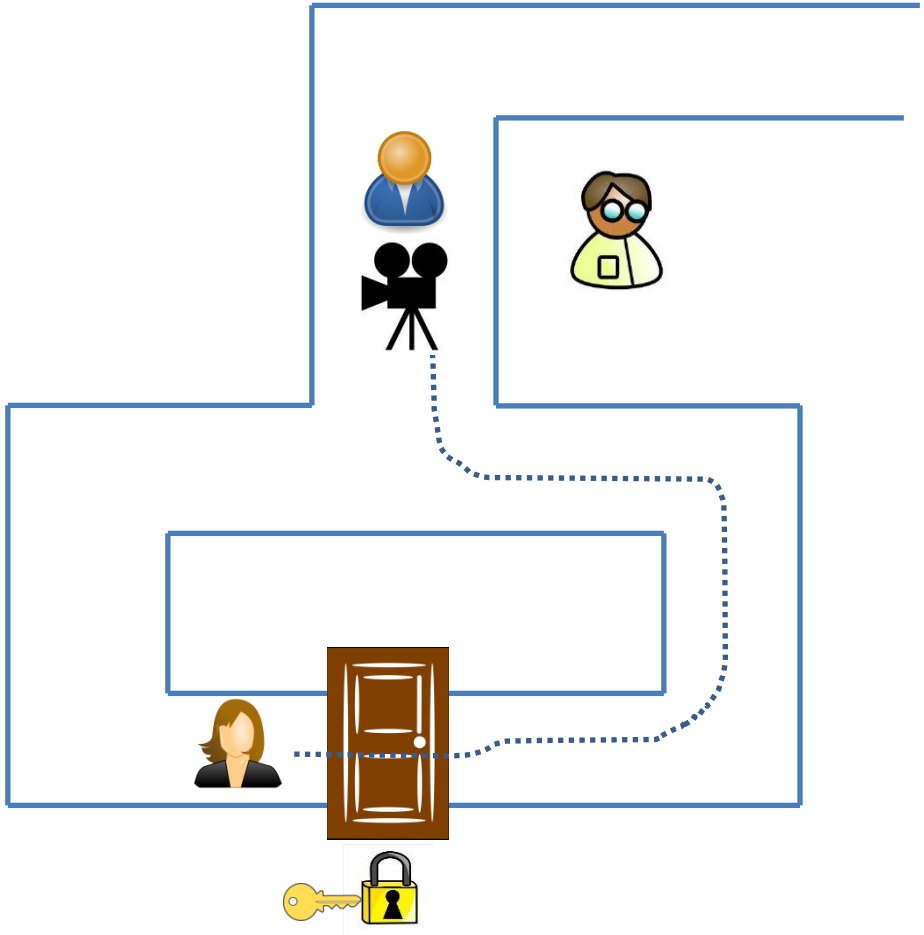
- Bob flips a coin
  - Heads = Left
  - Tails = Right
- Bob then yells down the hallway and demands that Alice appear from that side
- If Alice is already on the same side she simply walks out
- If Alice is on the wrong side she needs to unlock the door

ZK Proof Example



- Is this sound? Can Alice prove false statements to Bob?
- Is this complete? Will Bob always accept true statements?
- Is this zero-knowledge? Does Bob learn anything other than the truth about whether or not Alice can unlock the door?

ZK Proof Example



- Can Bob convince Charlie that Alice knows how to unlock the door?
- If the proof fails, if Alice comes out from the wrong side, does this prove that Alice does not know how to unlock the door?

Zero Knowledge

- What does it mean to say that V does not learn any knowledge other than the truth of the statement being proven?
  - What is knowledge? – Hard question, will not attempt to answer
  - What does it mean to say that V gained no knowledge? – We will work with this
- What does it mean to say that V gained no knowledge?
  - V after executing the protocol cannot do anything that V could not already do, in particular we are talking about $V's$ ability to compute statements
  - Even the protocol generated by the proof interactions between V and P could have been generated by V
  - To prove that V gained no knowledge from the interaction, we construct an algorithm called a 'simulator' where V generates a transcript of the protocol that is indistinguishable from a real interaction with P

Zero Knowledge Proof of Equality of Discrete Logarithm

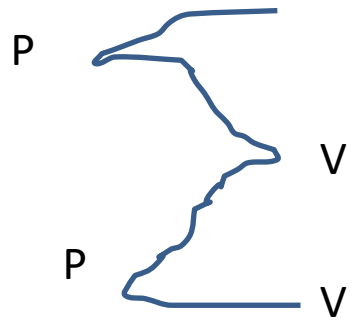
- P: Prover
- V: Verifier
- Common Input: $(g, g') \leftarrow_R \mathbb{Z}_q \times \mathbb{Z}_q$ generators, $(h, h') \leftarrow \mathbb{Z}_q \times \mathbb{Z}_q$

- P wants to convince V that it knows an $x \leftarrow \mathbb{Z}_q$ s. t. $h = g^x, h' = g'^x$
- P does not want V to learn the value of x or otherwise be able to compute it any easier because of their interaction

- We will use an interactive zero-knowledge protocol to prove this statement

Protocol Π For Proving Equality of Discrete Logarithm

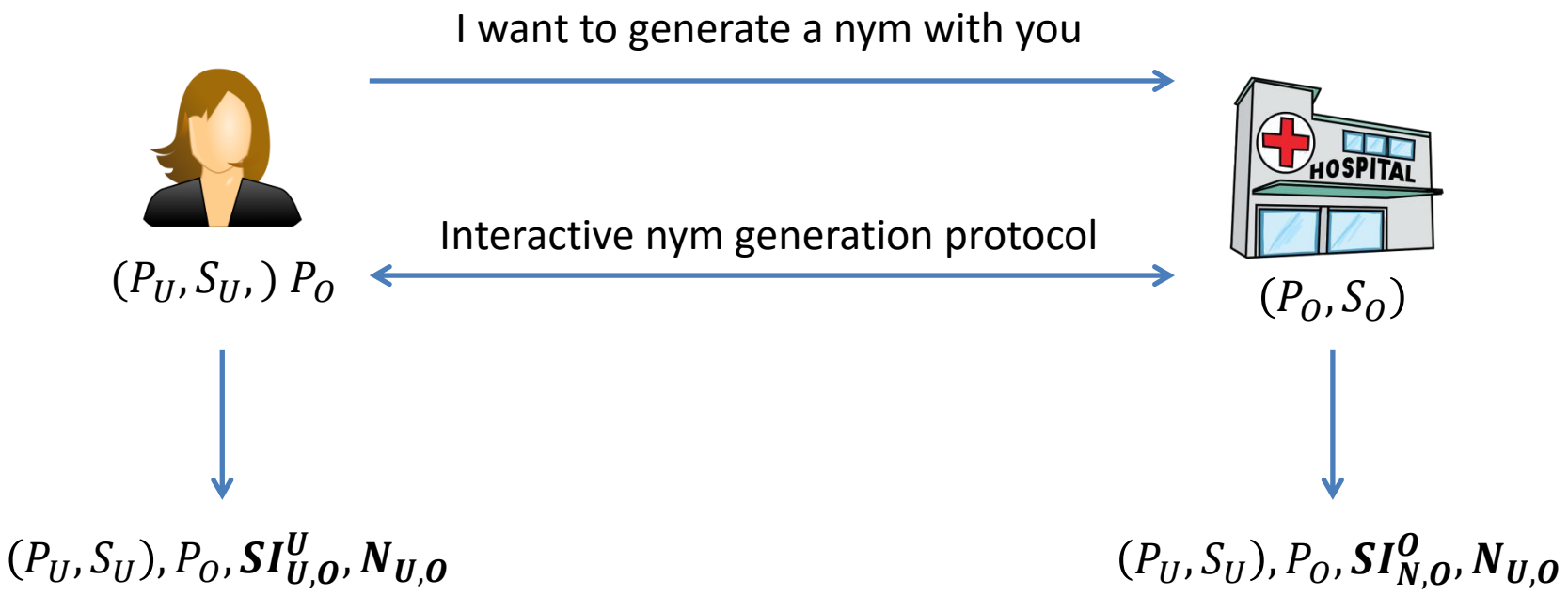
- $P \rightarrow V$: Choose $r \leftarrow_R \mathbb{Z}_q$, Send $(A = g^r, B = g'^r)$
- $V \rightarrow P$: Choose $c \leftarrow_R \mathbb{Z}_q$, Send c
- $P \rightarrow V$: Send y= $r + cx \ \ mod \ \ q$
- V: Check that $g^y = Ah^c$ and $g'^y = Bh'^c$



- Check:
- $g^y = g^{r+cx} = g^r g^{cx} = Ag^{cx} = Ah^c$
- $g'^y = g'^{r+cx} = g'^r g'^{cx} = Bg'^{cx} = Bh'^c$

- Is this sound?
- Is this complete?
- Is this zero-knowledge?
- If the prover showed this protocol to the verifier at a later date, would the verifier recognize it?
  - Produce a 'blinded' version of the protocol where it will not be recognized.

Back to Nym Generation



Nym Generation

- U: $(P_U, S_U), P_O = (g^x, x), g^y$
- O: $(P_O, S_O) = (g^y, y)$

- U: Choose $\gamma \leftarrow_R \mathbb{Z}_q$, Set $a' = g^\gamma, b' = a'^x$
- $U \rightarrow O$: Send (a', b')
- O: Choose $r \leftarrow_R \mathbb{Z}_q$, Set $a = a'^r$
- $O \rightarrow U$: Send a
- U: Compute $b = a^x$
- $U \longleftrightarrow O$: Execute Π to show that $\log_a b = log_{a'} b'$
- U, O: Remember $U's$ nym as $N = (a, b)$

Issue Credential

- $U: (P_U, S_U), P_O = (g^x, x), g^y, N_{U,O} = (a, b = a^x) = (\ g^{\gamma r}, g^{\gamma r x})$
- $O: (P_O, S_O) = (g^y, y), N_{U,O} = (a, b)$, Public Credential Key: $(g, h_1, = g^{s_1}, h_2 = g^{s_2})$, Secret Credential Key: (s_1, s_2)

- $O \rightarrow U$: Send $(A = b^{s_2}, B = (ab^{s_2})^{s_1})$
- U: Choose $\gamma \leftarrow_R \mathbb{Z}_q$
- $O \longleftrightarrow U$: Run Γ to show $\log_b A = \log_g h_2$ with verifier input γ, Obtain transcript T_1
- $O \longleftrightarrow U$: Run Γ to show $\log_{(a,A)} B = \log_g h_1$ with verifier input γ, Obtain transcript T_2
- U: Remember credential $C_{U,O} = (a^\gamma, b^\gamma, A^\gamma, B^\gamma, T_1, T_2)$

Transfer Credential

- $O's$ public credential keys: $(g, h_1 = g^{s_1}, h_2 = g^{s_2})$
- $U's$ nym with O': (a'', b'') where $b'' = a''^{x}$
- User's credential from O: $C_{U,O} = (a', b', A', B', T_1, T_2)$

- O': Verify correctness of T_1 and T_2 as transcripts for Π_{NI} for showing $\log_{b'} A' = \log_g h_2$ and $\log_{a'A'} B' = \log_g h_1$

- $U \longleftrightarrow O'$: Execute protocol Π to show $\log_{a'} b' = \log_a b$

Single-Use / Multiple-Use Credentials

- Single Use Credential: May safely be used once, but if used more than once, it would allow the user's nyms to be linked together

- Multiple-use Credential: May safely be used unlimited times without allowing the user's nyms to be linked

- K-Use Credentials? Can you create a credential that can be used a finite number of times before being able to link together a user's nyms?
  - Yes but its hard and very complicated

Expiration Date

- Add a date field into the non-interactive proof protocol such that the verifier only accepts if the current date is less than the expiration date

- Also needs to add corresponding fields into the credential and the corresponding machinery when verifying the credential

Credential Revocation

- This is going to require a trusted third party like the CA
- Revocations would have to be input with the CA, then
- When a credential is used, before it is verified, the organization will check with the CA to see if the credential has been revoked

Are there other problems here?

(aLiCe, Birthdate, Patient File, …)

(yasuo_only, Birthdate, Prescriptions, …)

Prescription for aLiCe

I don't know who aLiCe is, you are yasuo_only

{aLiCe, yasuo_only}

Credentials for a Review System?