

Efficient Private Techniques for Verifying Social Proximity

Antonio Nicolosi

(based on joint work w/ M. Freedman, M. Kaminsky, B. Karp, M. Ryan)



LSS Lunch Meeting

Oct 23rd, 2007

Motivation:

Spam Filtering via Social Networks

- Spam Filtering as an **authentication** problem
 - Legitimate senders authenticate as non-spammers
 - Recipients aggressively filter “unauthenticated” e-mails
- Authentication \approx Short **sender–recipient** chain
 - Will focus on 2-hop chains
- How can senders demonstrate such chains?

Demonstrating Social Proximity: Strawman

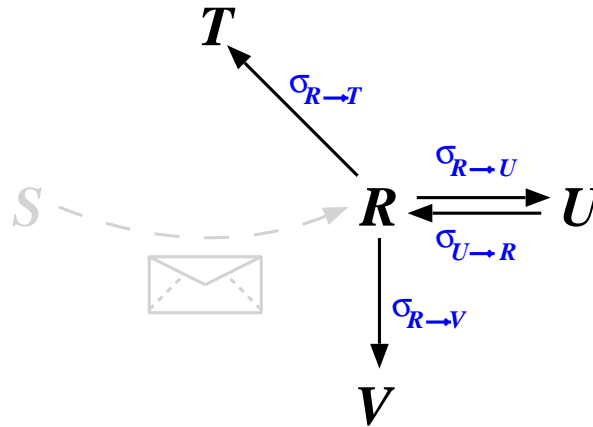
S

R

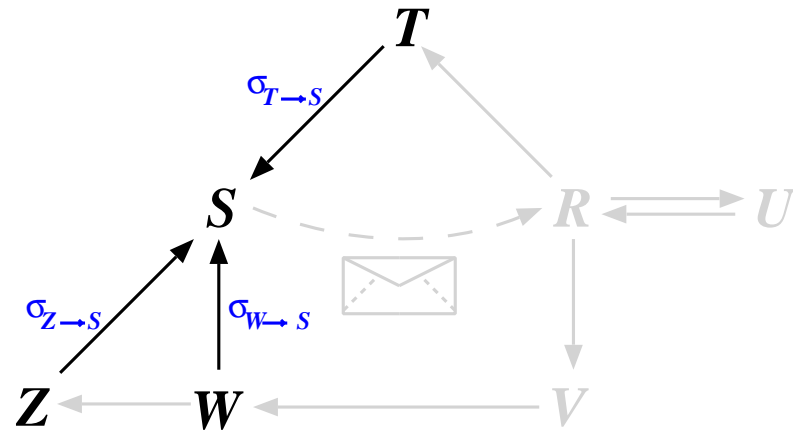
Demonstrating Social Proximity: Strawman



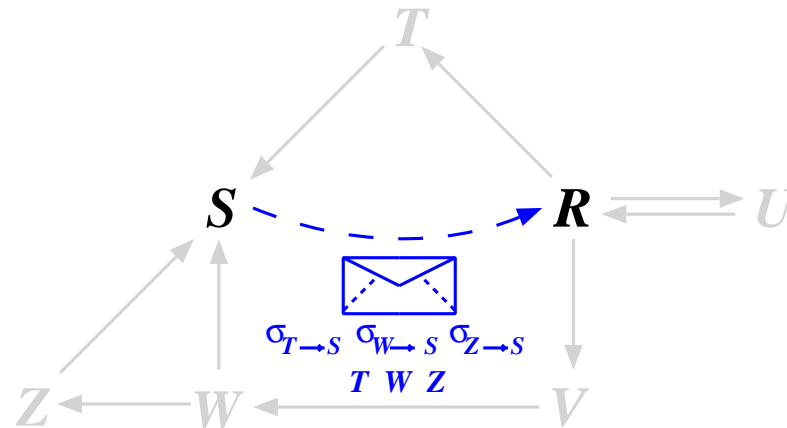
Demonstrating Social Proximity: Strawman



Demonstrating Social Proximity: Strawman

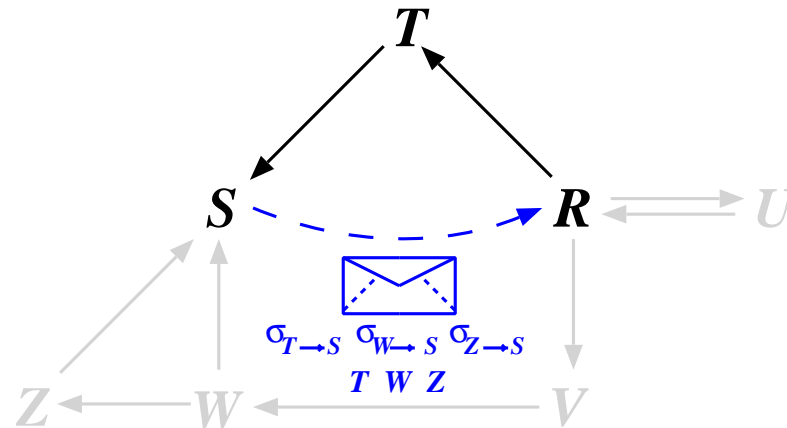


Demonstrating Social Proximity: Strawman



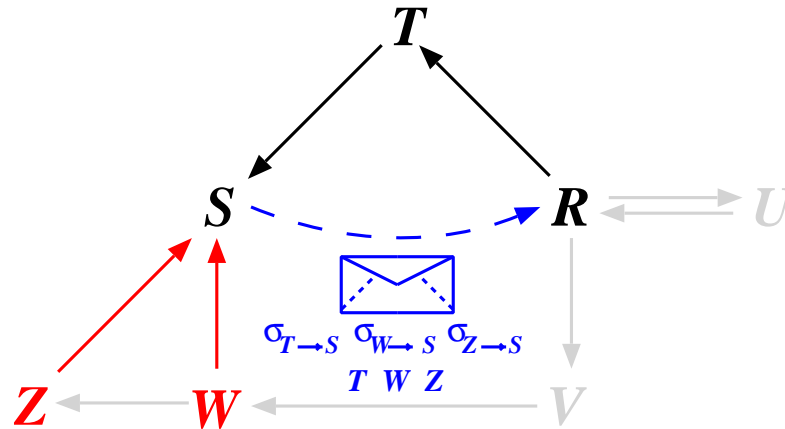
- S sends R a list of his contacts

Demonstrating Social Proximity: Strawman



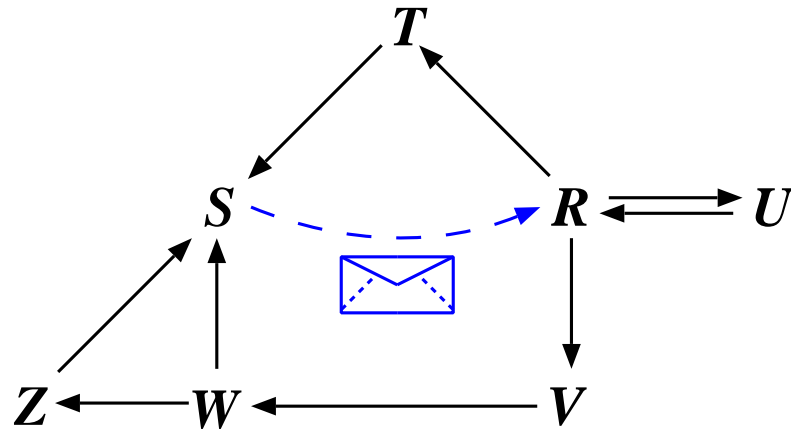
- S sends R a list of his contacts
 - R learns about a **bridging** friend, T

Demonstrating Social Proximity: Strawman



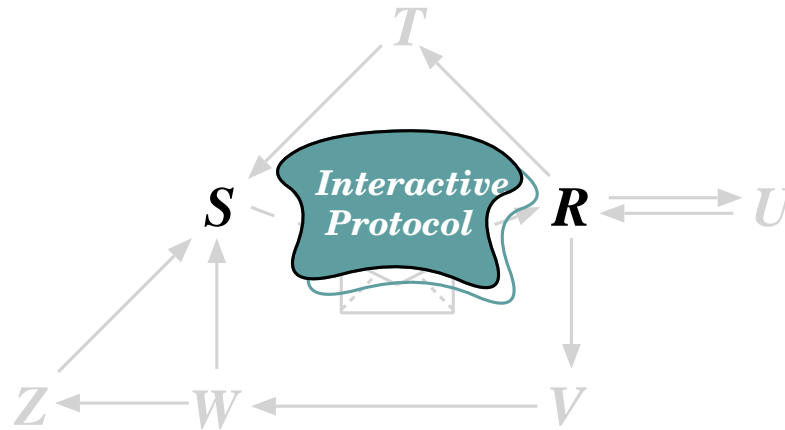
- S sends R a list of his contacts
 - R learns about a **bridging** friend, T
- **Privacy concerns**
 - Discloses more information than needed

Demonstrating Social Proximity: RE:



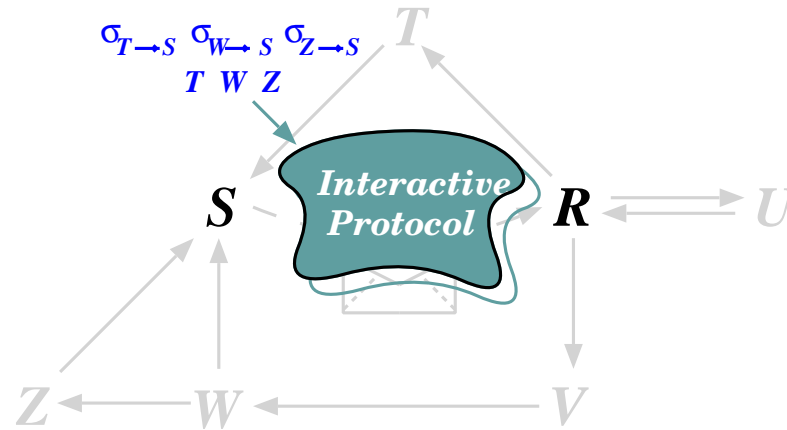
- Use interactive protocol for private set intersection

Demonstrating Social Proximity: **RE:**



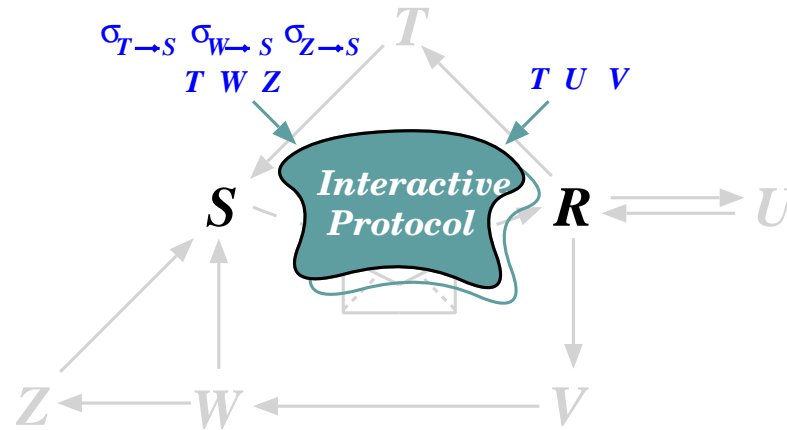
- Use interactive protocol for private set intersection

Demonstrating Social Proximity: RE:



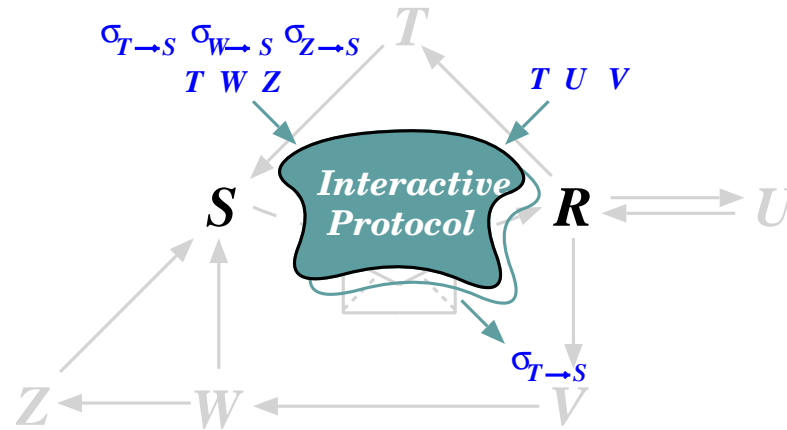
- Use interactive protocol for private set intersection

Demonstrating Social Proximity: **RE:**



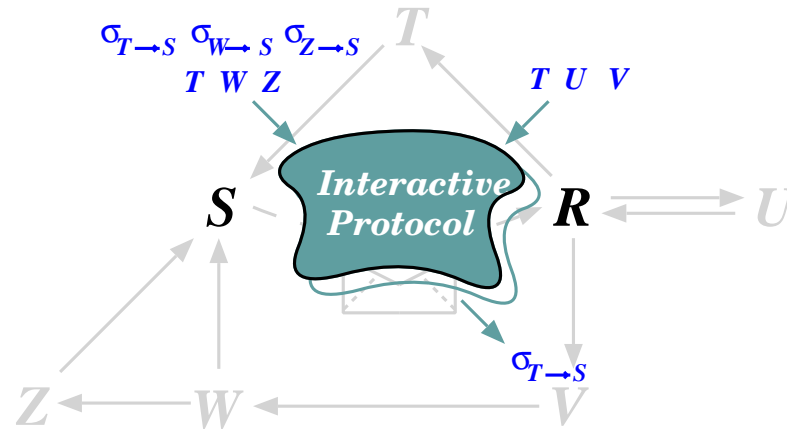
- Use interactive protocol for private set intersection

Demonstrating Social Proximity: **RE**:



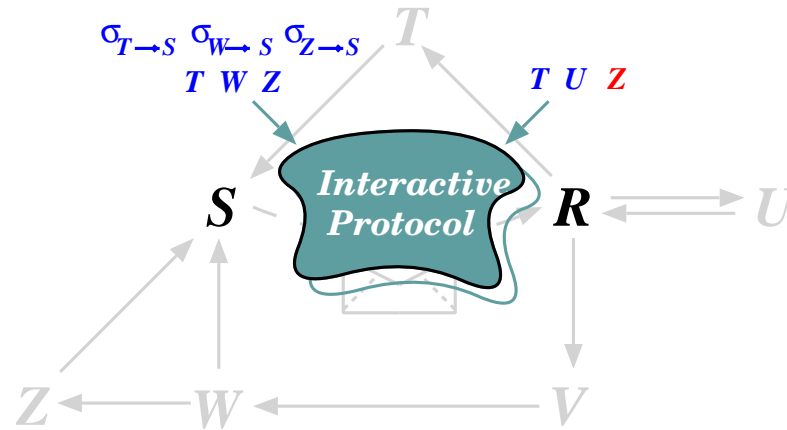
- Use interactive protocol for private set intersection
- R can't learn anything but their bridging friends

Demonstrating Social Proximity: **RE:**



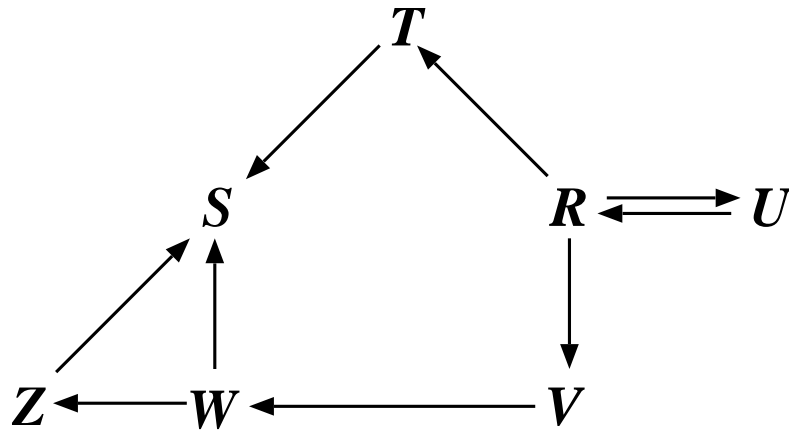
- Use interactive protocol for private set intersection
- R can't learn anything but their bridging friends
- Or can she?!
 - Multi-party computation does not prevent lying

Demonstrating Social Proximity: **RE:**



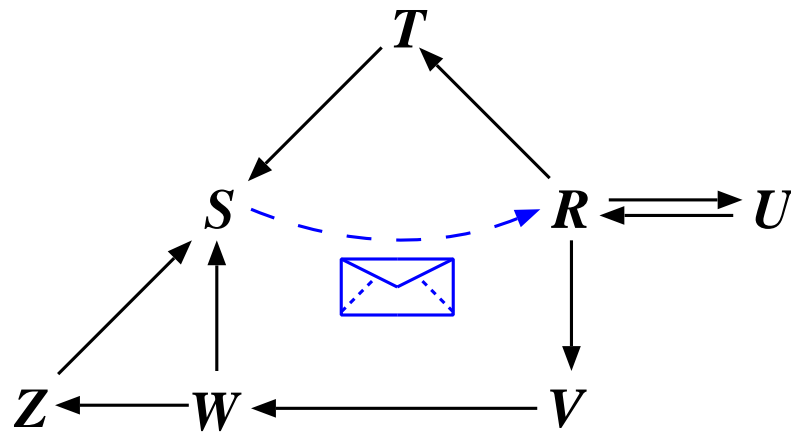
- Use interactive protocol for private set intersection
- R can't learn anything but their bridging friends
- Or can she?!
 - Multi-party computation does not prevent lying

Social Networks and Proximity Checks



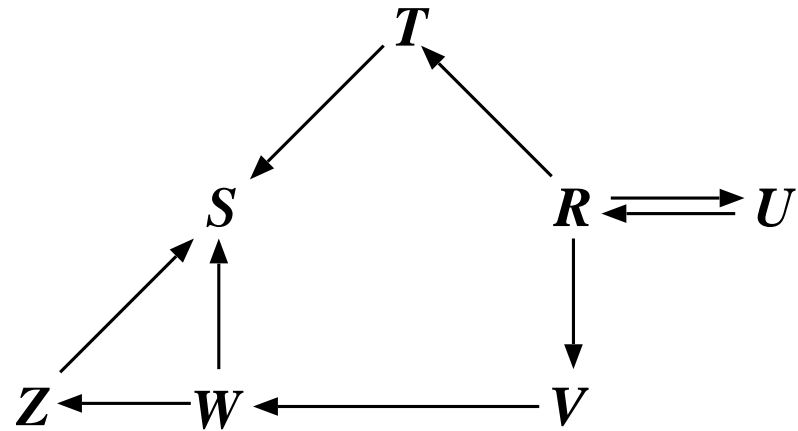
- Social relationships represented as directed edges

Social Networks and Proximity Checks



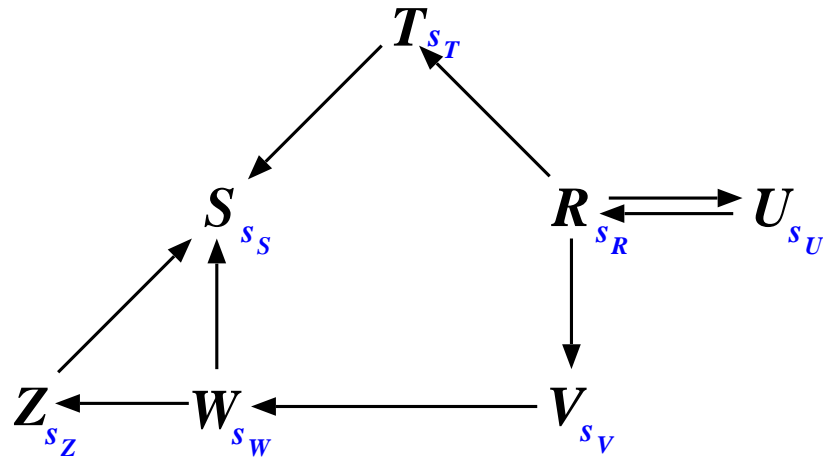
- Social relationships represented as directed edges
- **Proximity Check**
 - S can help R finding out about “bridging friends”, *i.e.*, users like T : $R \rightarrow T \rightarrow S$

A Practical Hash-Based Solution



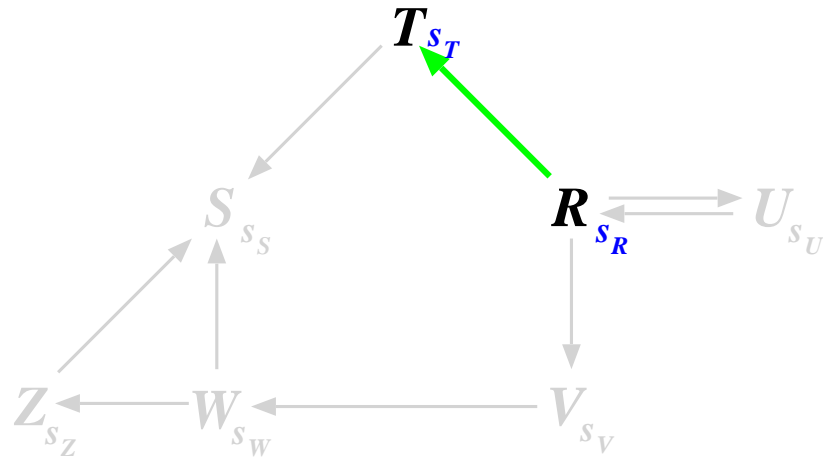
- Each user keeps
 - A PK pair for signatures (not shown)

A Practical Hash-Based Solution

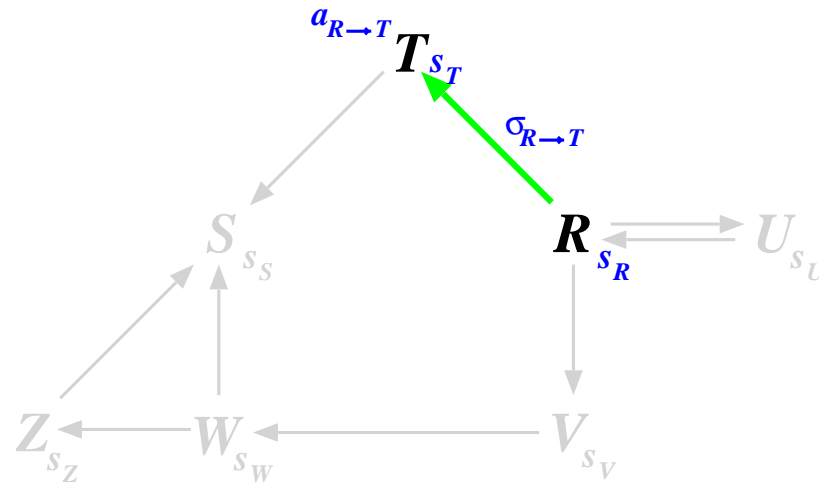


- Each user keeps
 - A **PK pair** for signatures (not shown)
 - A **master seed** from which shared secrets will be derived

A Practical Hash-Based Solution (cont'd)

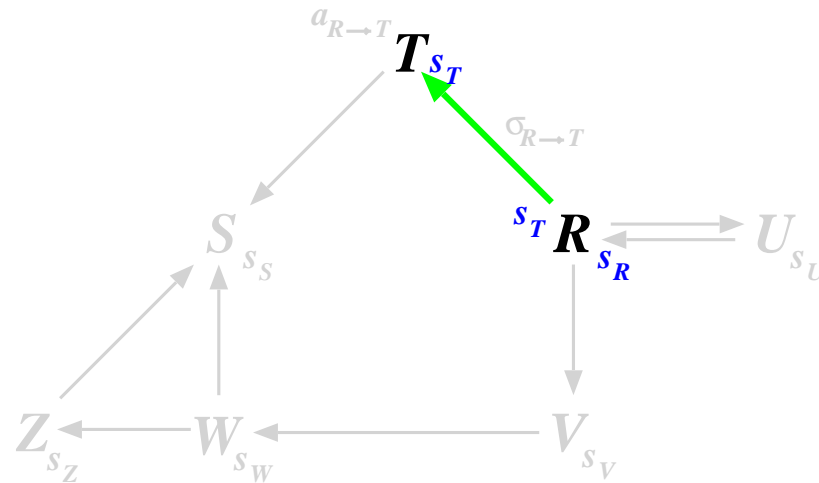


A Practical Hash-Based Solution (cont'd)



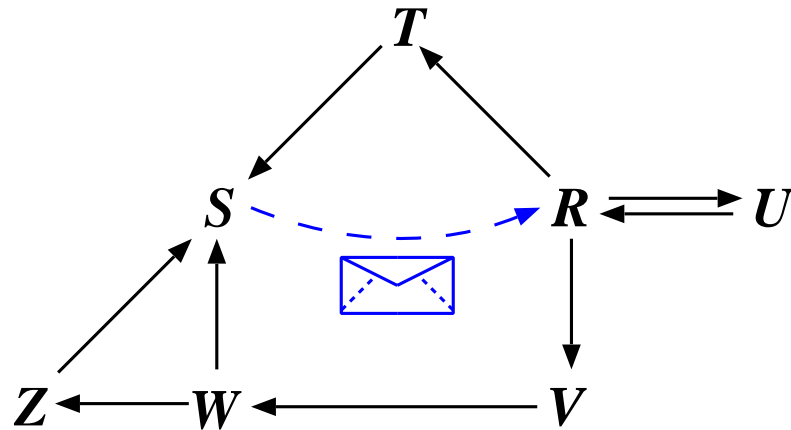
- [Forward trust] R gives T :
 - Signed attestation $\sigma_{R \rightarrow T}$
 - Shared secret $a_{R \rightarrow T} = \mathcal{H}(s_R, R, T)$

A Practical Hash-Based Solution (cont'd)

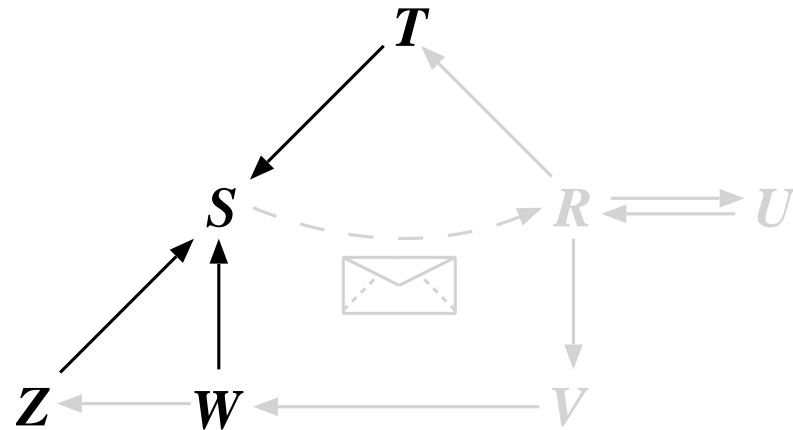


- **[Forward trust] R gives T :**
 - Signed attestation $\sigma_{R \rightarrow T}$
 - Shared secret $a_{R \rightarrow T} = \mathcal{H}(s_R, R, T)$
- **[Backward authorization] T gives R :**
 - Master seed s_T

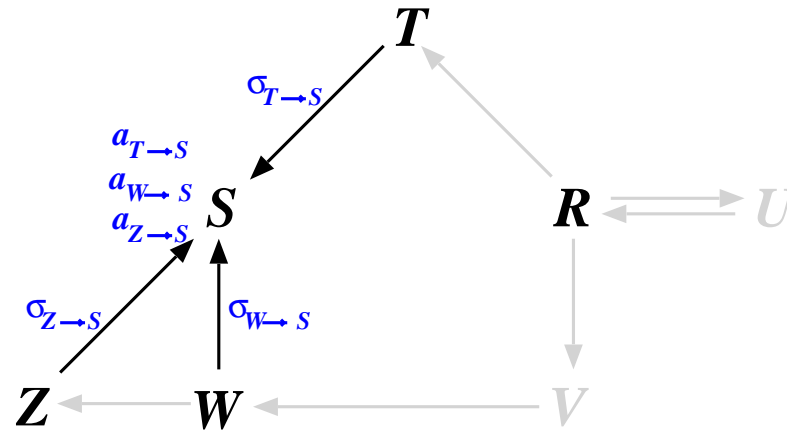
Implementing Proximity Checks—Warmup



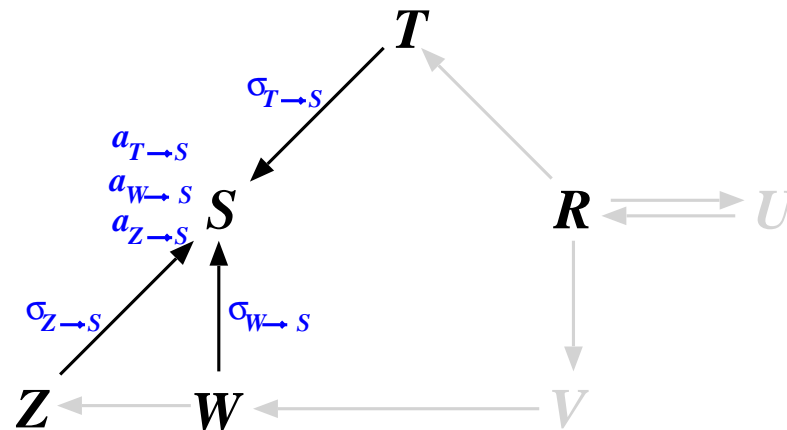
Implementing Proximity Checks—Warmup



Implementing Proximity Checks—Warmup



Implementing Proximity Checks—Warmup

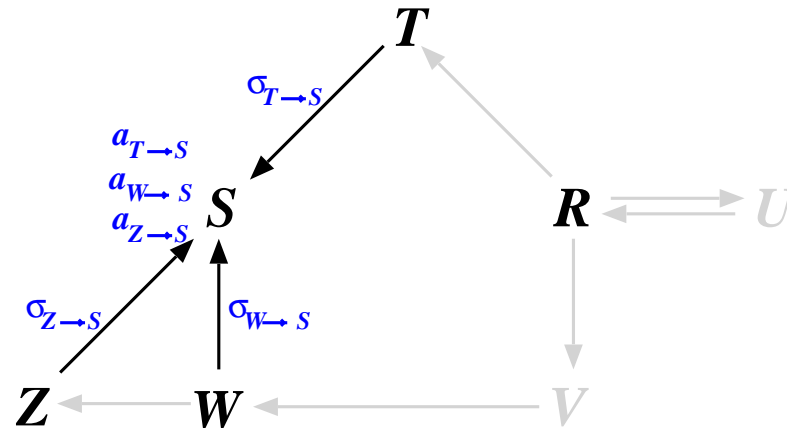


- **Sender's Computations:**

User S

$\{\sigma_{T \rightarrow S}\}_{a_{T \rightarrow S}}$

Implementing Proximity Checks—Warmup

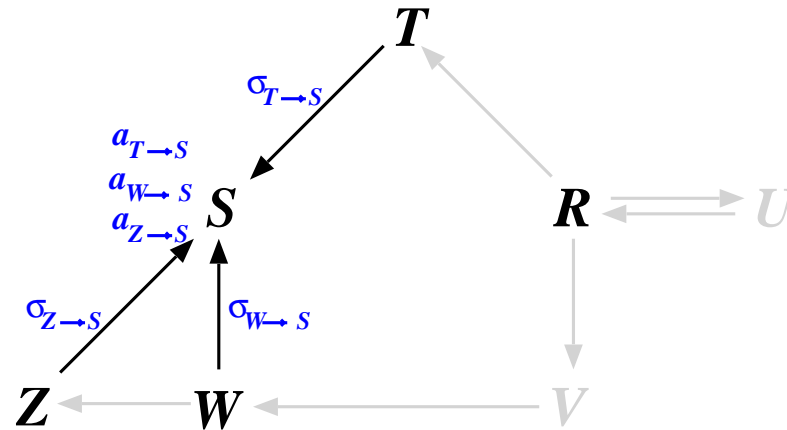


- **Sender's Computations:**

User S

$\{\sigma_{T \rightarrow S}\}_{a_{T \rightarrow S}}$
$\{\sigma_{W \rightarrow S}\}_{a_{W \rightarrow S}}$

Implementing Proximity Checks—Warmup

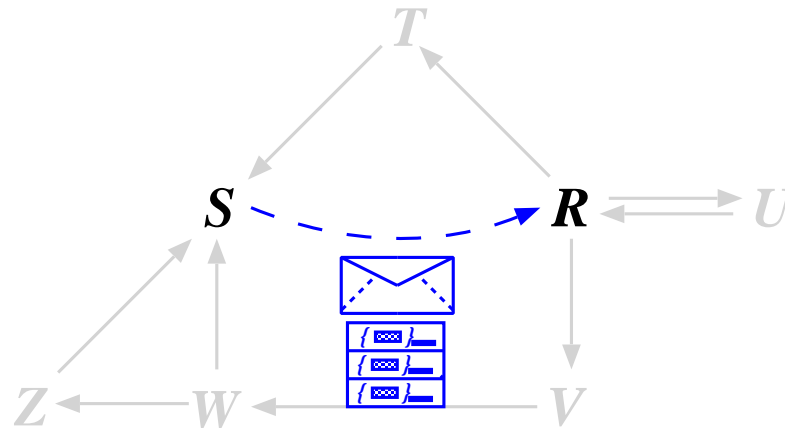


- **Sender's Computations:**

User S

$\{\sigma_{T \rightarrow S}\}_{a_{T \rightarrow S}}$
$\{\sigma_{W \rightarrow S}\}_{a_{W \rightarrow S}}$
$\{\sigma_{Z \rightarrow S}\}_{a_{Z \rightarrow S}}$

Implementing Proximity Checks—Warmup

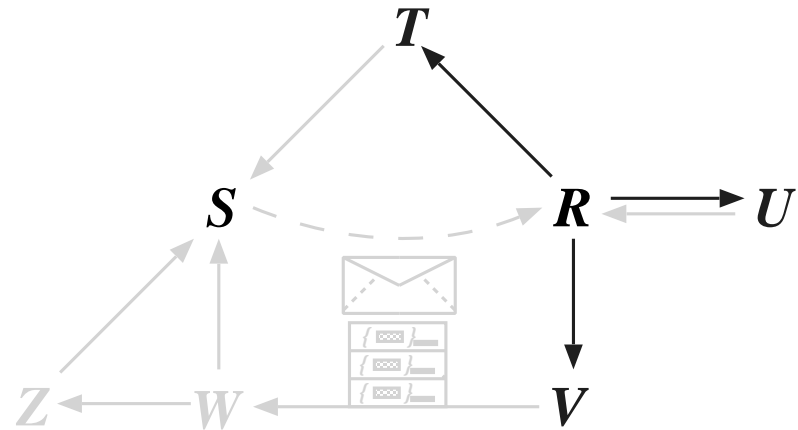


- **Sender's Computations:**

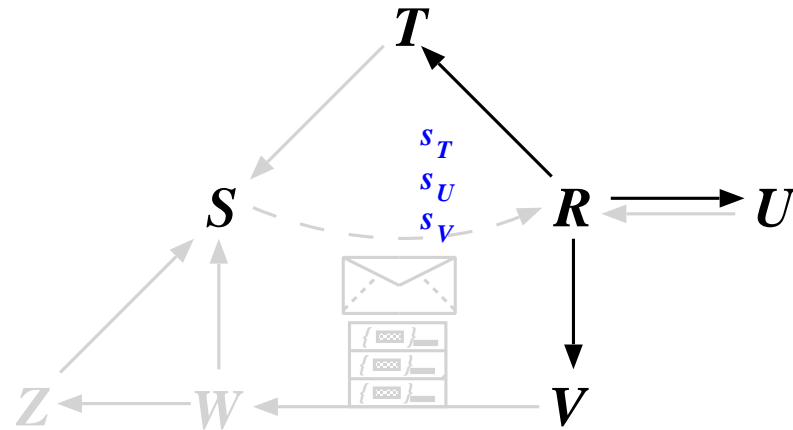
User S

$\{\sigma_{T \rightarrow S}\}_{a_{T \rightarrow S}}$
$\{\sigma_{W \rightarrow S}\}_{a_{W \rightarrow S}}$
$\{\sigma_{Z \rightarrow S}\}_{a_{Z \rightarrow S}}$

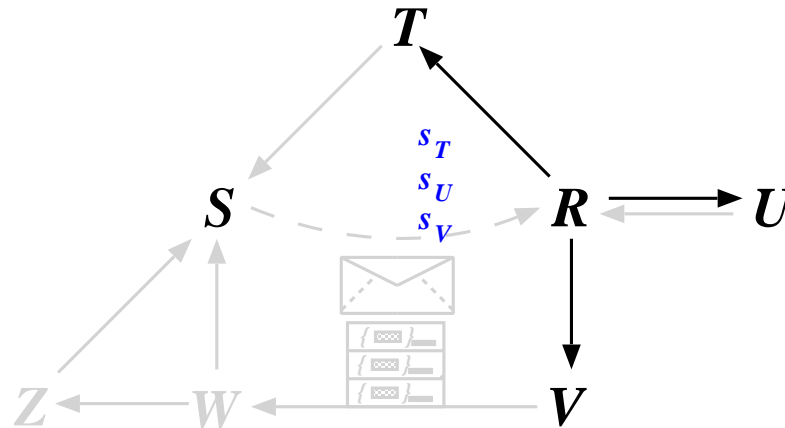
Implementing Proximity Checks—Warmup



Implementing Proximity Checks—Warmup



Implementing Proximity Checks—Warmup

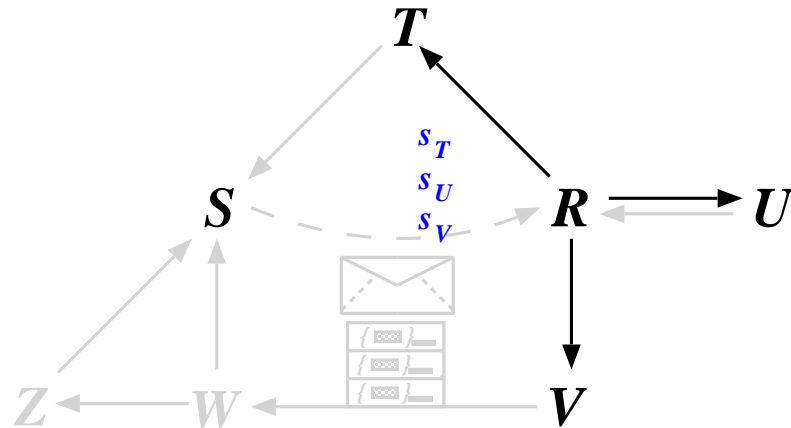


- Receiver's Computations:

User R

$$\boxed{a_{T \rightarrow S}} = \mathcal{H}(s_T, T, S)$$

Implementing Proximity Checks—Warmup

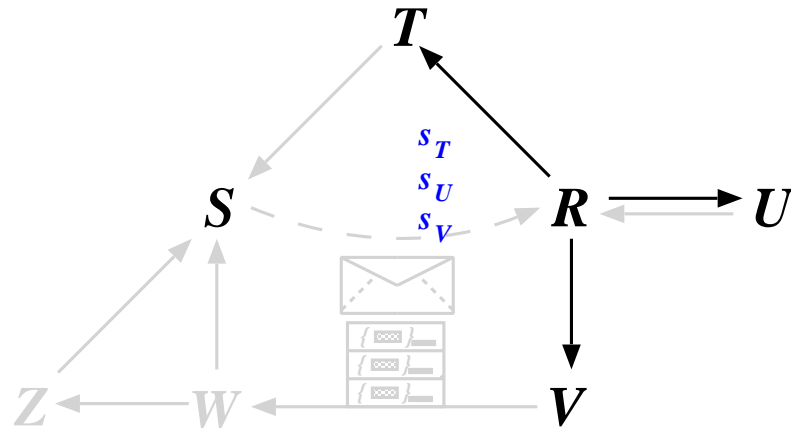


- Receiver's Computations:

User R

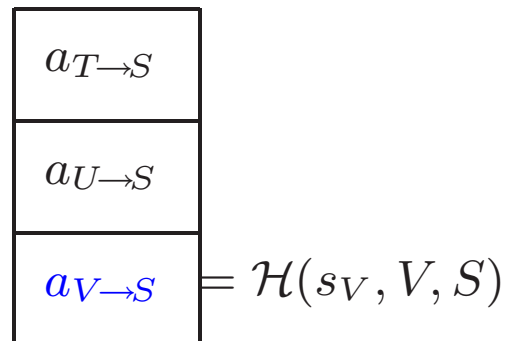
$a_{T \rightarrow S}$
$a_{U \rightarrow S} = \mathcal{H}(s_U, U, S)$

Implementing Proximity Checks—Warmup

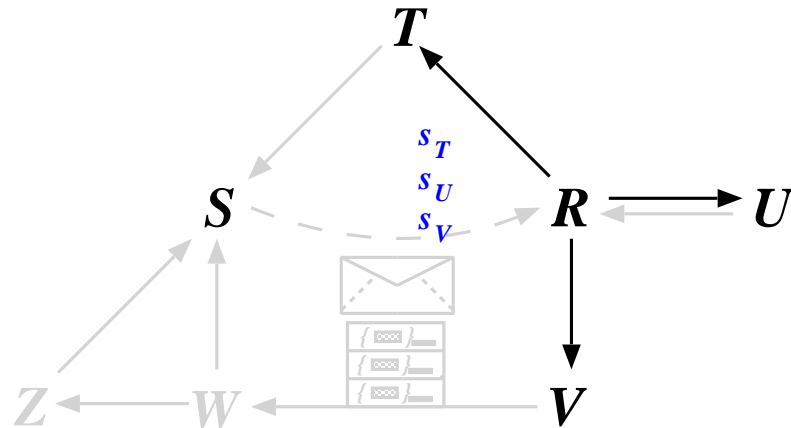


- Receiver's Computations:

User R



Implementing Proximity Checks—Warmup

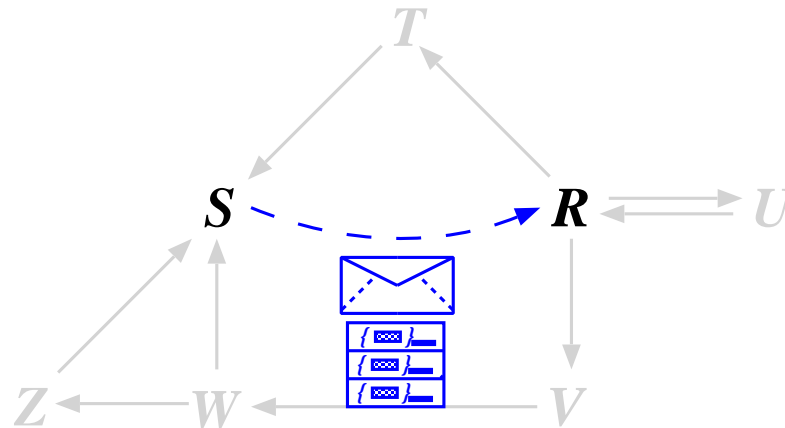


- Receiver's Computations:

User R

$a_{T \rightarrow S}$
$a_{U \rightarrow S}$
$a_{V \rightarrow S}$

Implementing Proximity Checks—Warmup



- Receiver's Computations:

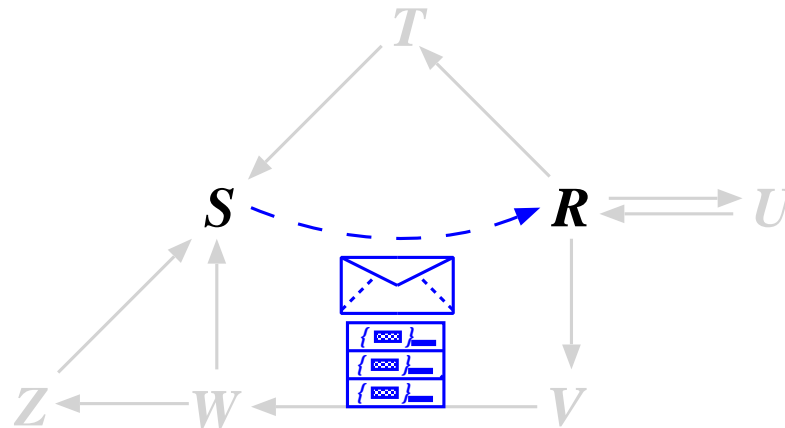
User S

$\{\sigma_{T \rightarrow S}\} a_{T \rightarrow S}$
$\{\sigma_{W \rightarrow S}\} a_{W \rightarrow S}$
$\{\sigma_{Z \rightarrow S}\} a_{Z \rightarrow S}$

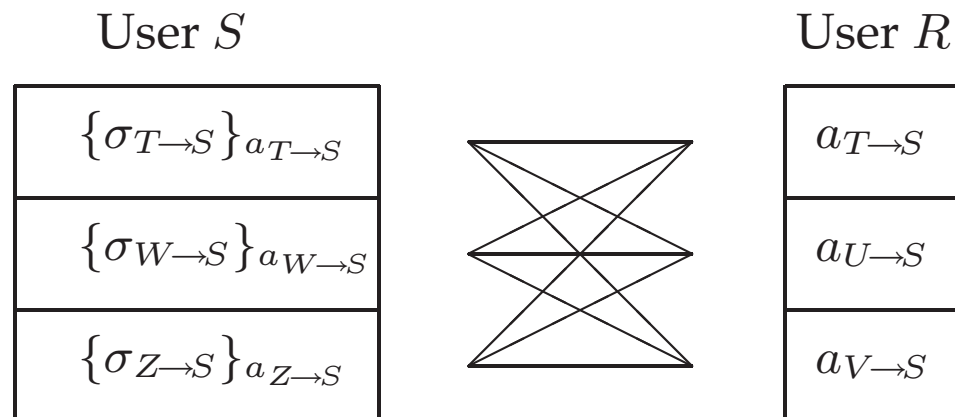
User R

$a_{T \rightarrow S}$
$a_{U \rightarrow S}$
$a_{V \rightarrow S}$

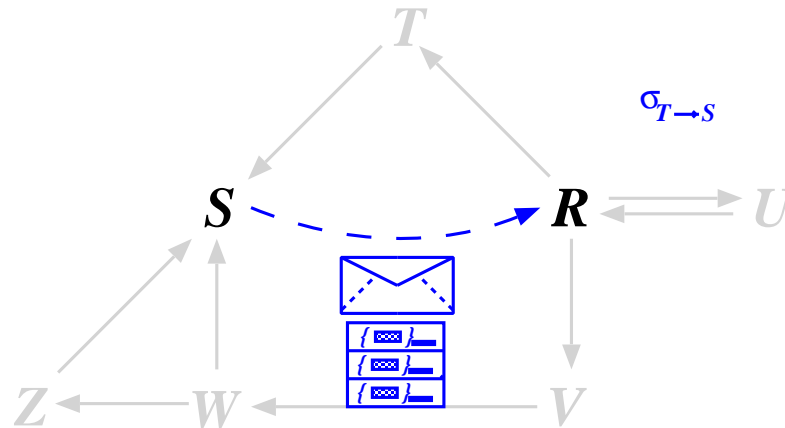
Implementing Proximity Checks—Warmup



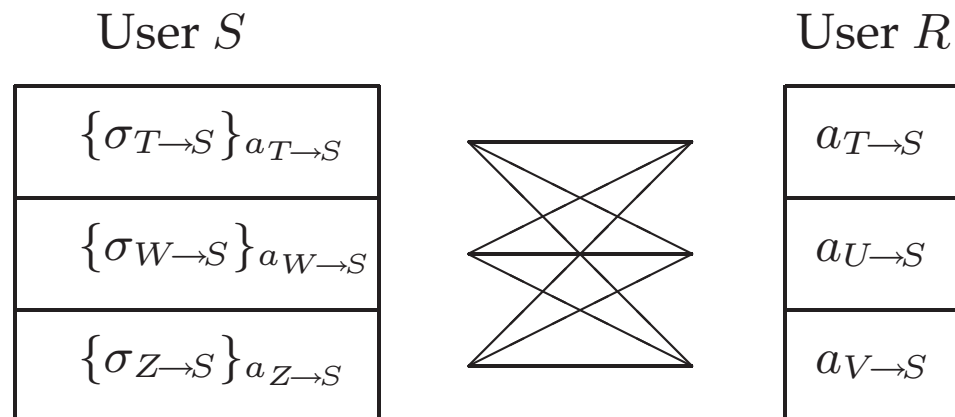
- Receiver's Computations:



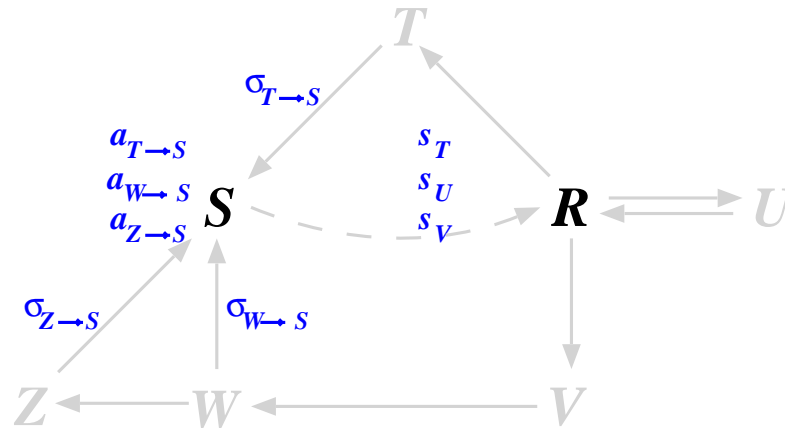
Implementing Proximity Checks—Warmup



- Receiver's Computations:



Speeding up Proximity Checks

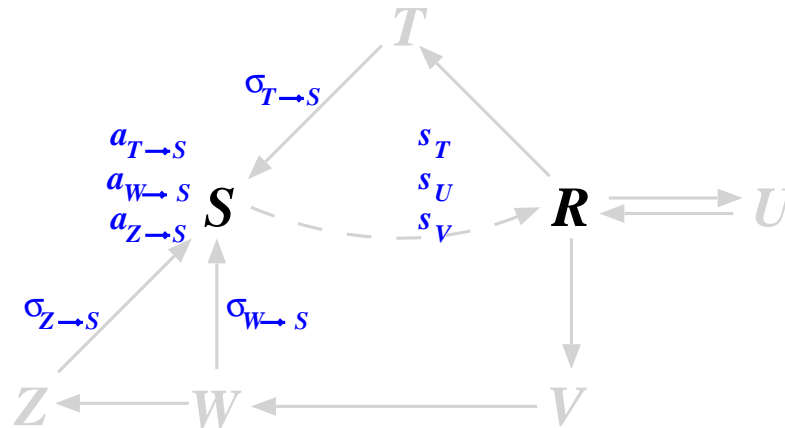


- Idea: Make entries recognizable adding MAC's

User S

$\{\sigma_{W \rightarrow S}\} a_{W \rightarrow S}$
$\{\sigma_{Z \rightarrow S}\} a_{Z \rightarrow S}$
$\{\sigma_{T \rightarrow S}\} a_{T \rightarrow S}$

Speeding up Proximity Checks

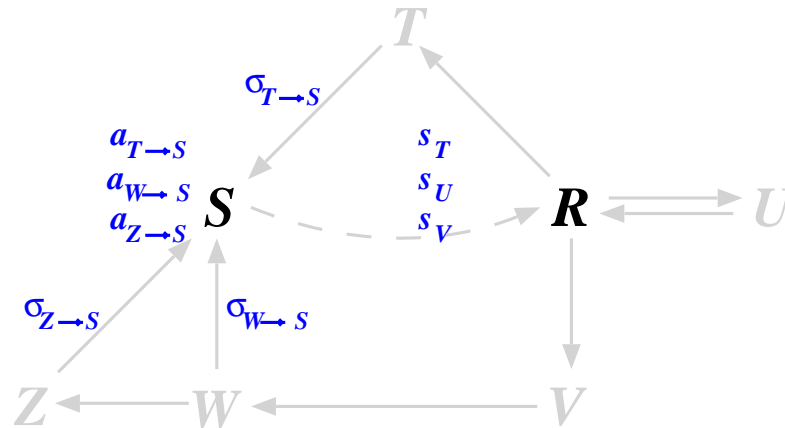


- **Idea: Make entries recognizable adding MAC's**

User S

$\{\sigma_{W \rightarrow S}\} a_{W \rightarrow S}$	$t_{W \rightarrow S}$
$\{\sigma_{Z \rightarrow S}\} a_{Z \rightarrow S}$	$t_{Z \rightarrow S}$
$\{\sigma_{T \rightarrow S}\} a_{T \rightarrow S}$	$t_{T \rightarrow S}$

Speeding up Proximity Checks



- Idea: Make entries recognizable adding MAC's

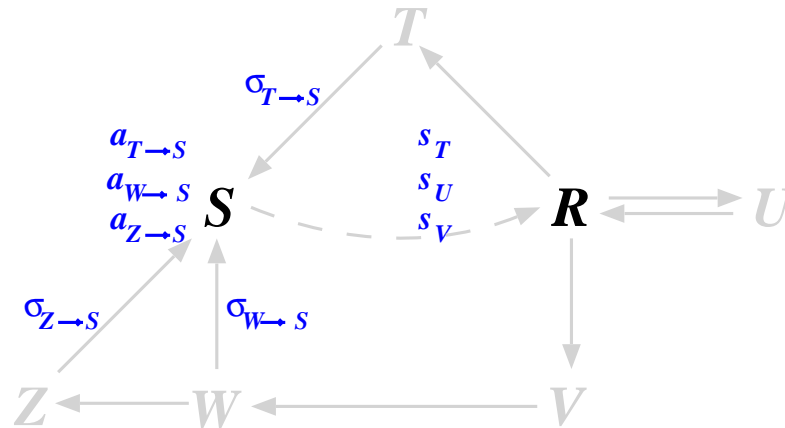
User S

$\{\sigma_{W \rightarrow S}\} a_{W \rightarrow S}$	$t_{W \rightarrow S}$
$\{\sigma_{Z \rightarrow S}\} a_{Z \rightarrow S}$	$t_{Z \rightarrow S}$
$\{\sigma_{T \rightarrow S}\} a_{T \rightarrow S}$	$t_{T \rightarrow S}$

User R

$a_{T \rightarrow S}$
$a_{U \rightarrow S}$
$a_{V \rightarrow S}$

Speeding up Proximity Checks



- Idea: Make entries recognizable adding MAC's

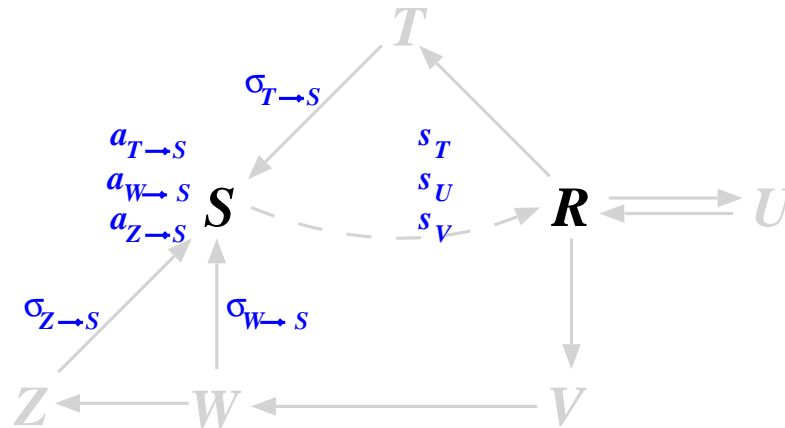
User S

$\{\sigma_{W \rightarrow S}\} a_{W \rightarrow S}$	$t_{W \rightarrow S}$
$\{\sigma_{Z \rightarrow S}\} a_{Z \rightarrow S}$	$t_{Z \rightarrow S}$
$\{\sigma_{T \rightarrow S}\} a_{T \rightarrow S}$	$t_{T \rightarrow S}$

User R

$t_{T \rightarrow S}$	$a_{T \rightarrow S}$
$t_{U \rightarrow S}$	$a_{U \rightarrow S}$
$t_{V \rightarrow S}$	$a_{V \rightarrow S}$

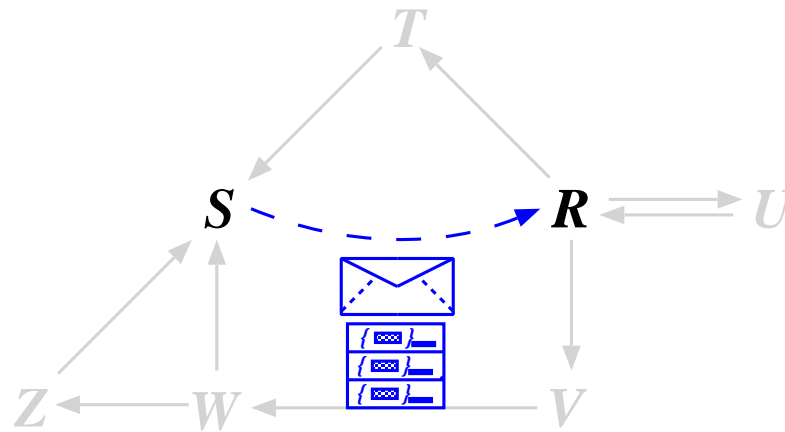
Speeding up Proximity Checks



- Idea: Make entries recognizable adding MAC's

User S	User R
$\{\sigma_{W \rightarrow S}\} a_{W \rightarrow S}$	$t_{W \rightarrow S}$
$\{\sigma_{Z \rightarrow S}\} a_{Z \rightarrow S}$	$t_{Z \rightarrow S}$
$\{\sigma_{T \rightarrow S}\} a_{T \rightarrow S}$	$t_{V \rightarrow S}$
	$t_{U \rightarrow S}$
	$a_{T \rightarrow S}$
	$a_{U \rightarrow S}$
	$a_{V \rightarrow S}$

Privacy Guarantees



- “Overly curious” R can’t learn anything beyond:
 - Bridging friends and attestations $(T, \sigma_{T \rightarrow S})$
 - Total number of users that have attested to S (3)

Simulation Performance

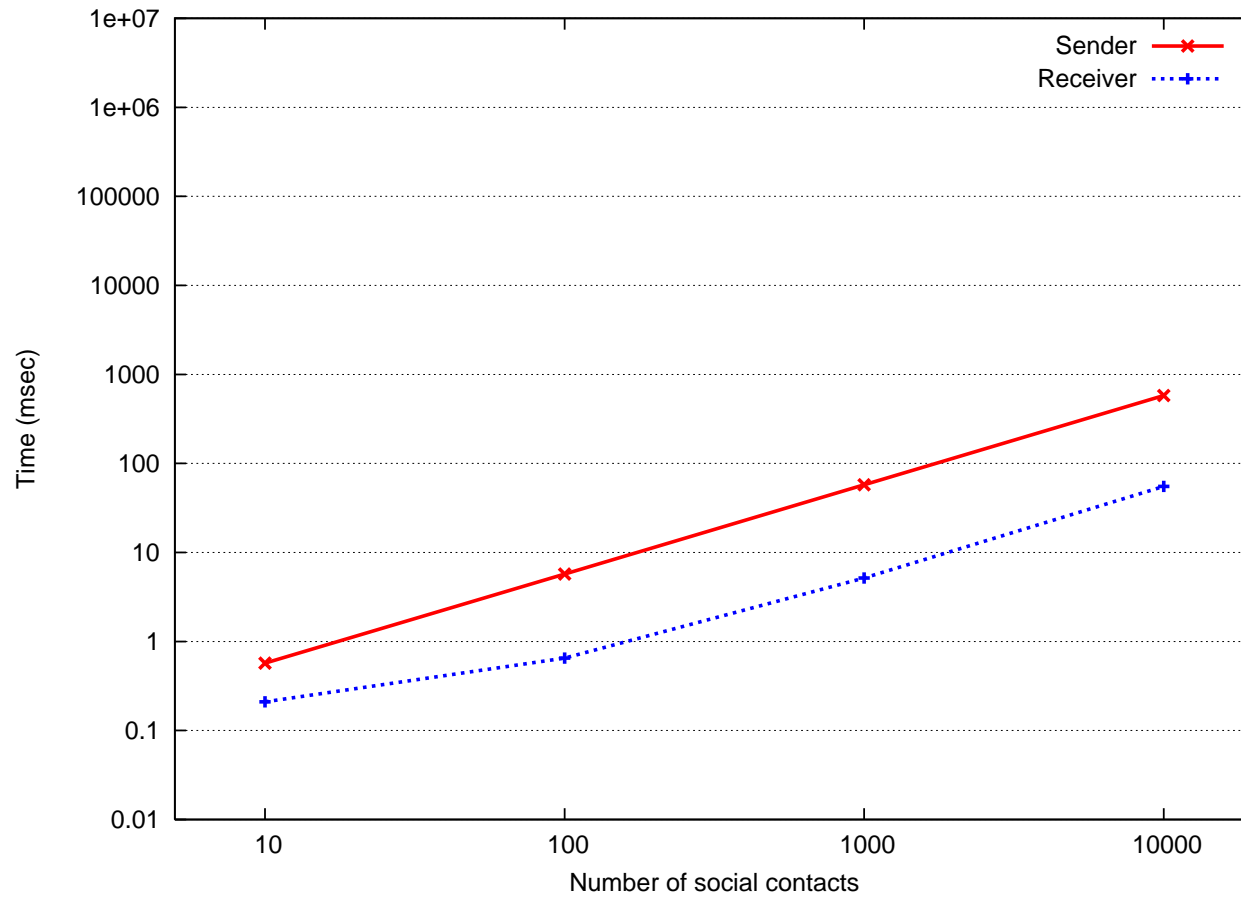
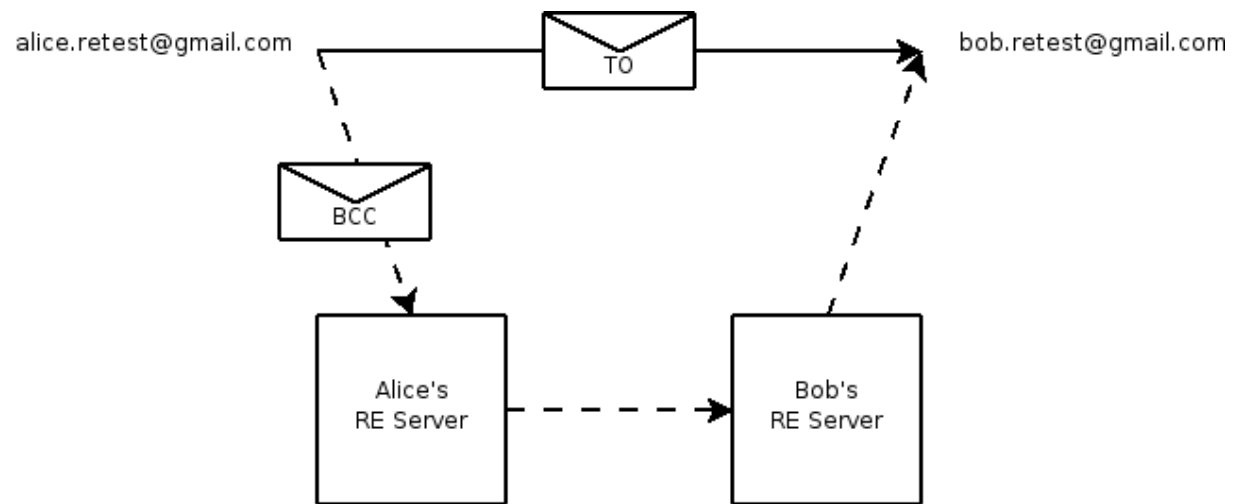


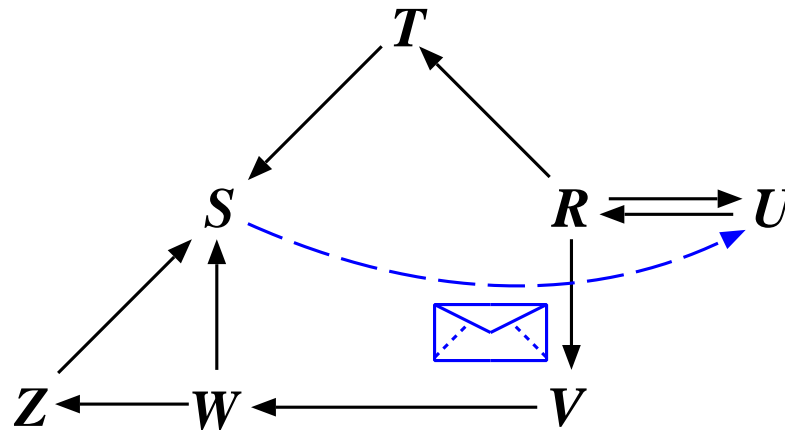
Table 1: Time (milliseconds) for proximity checks.

Integration with E-Mail Infrastructure



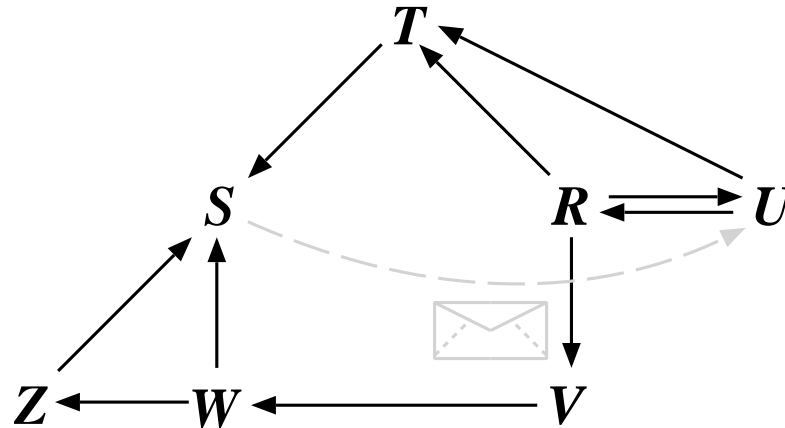
[This page intentionally left blank.]

Extension: Forward Security



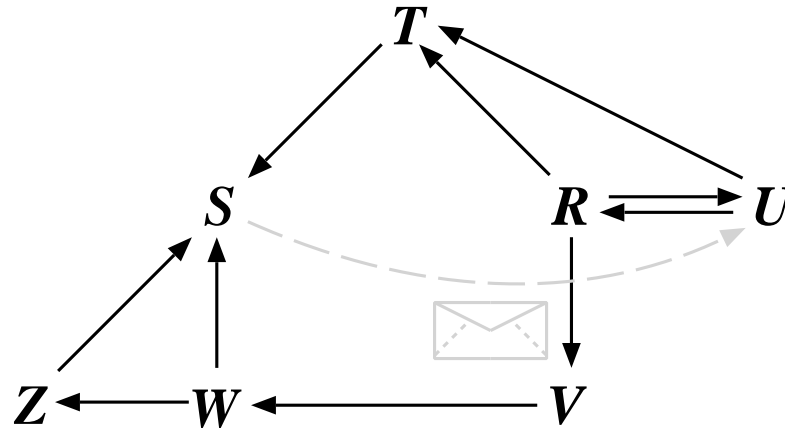
1. S e-mails U , but no bridging friend connects them

Extension: Forward Security



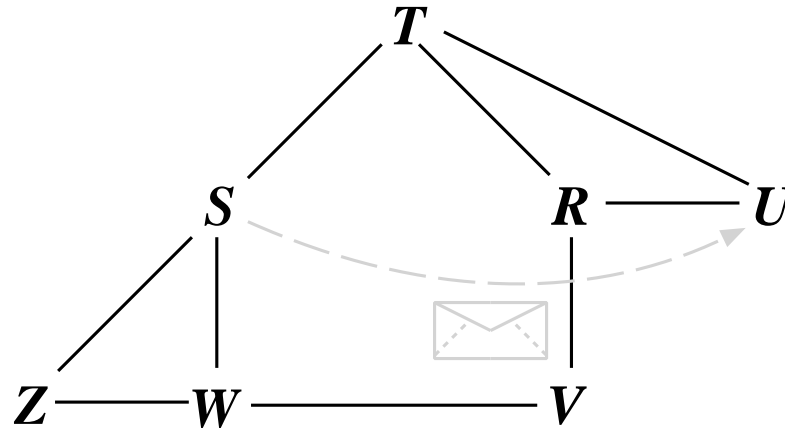
1. *S* e-mails *U*, but no bridging friend connects them
2. Later on, a new link arise from *U* to *T*

Extension: Forward Security



1. S e-mails U , but no bridging friend connects them
2. Later on, a new link arise from U to T
3. **Question**
 - Should U be able to find out that S was a friend T 's at 1.?

Extension: Forward Security

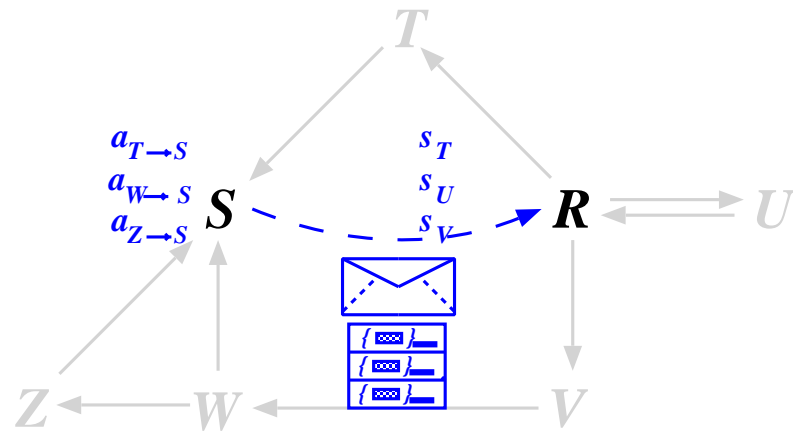


1. *S* e-mails *U*, but no bridging friend connects them
 2. Later on, a new link arise from *U* to *T*
 3. **Question**
 - Should *U* be able to find out that *S* was a friend *T*'s at 1.?
- Can prevent this using **hash chains**

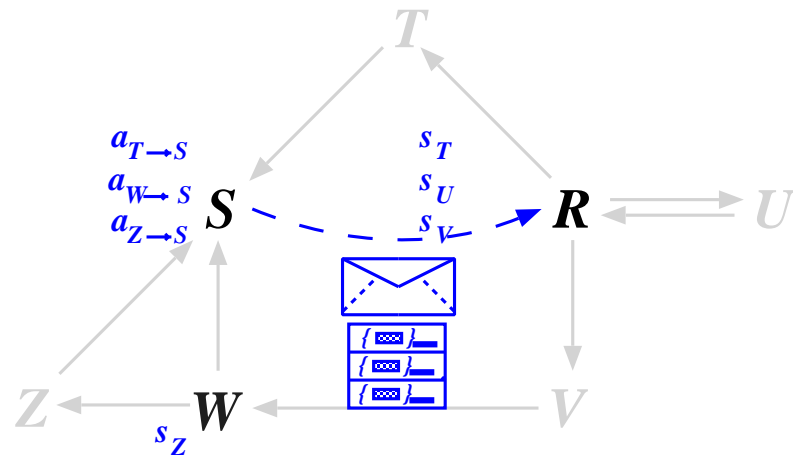
Privacy *vs.* Auditability

- **In principle, more “private” approach possible**
 - *R* only learns that a bridging friend **exists**
- **But no robustness to incorrect transitive predictions**
 - *“Which friends of mine vouched for this spammer?!”*

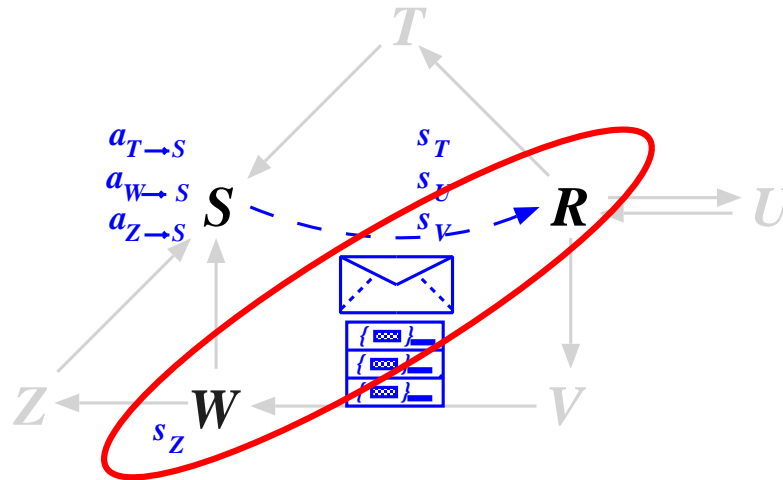
Privacy in the Face of Collusions



Privacy in the Face of Collusions

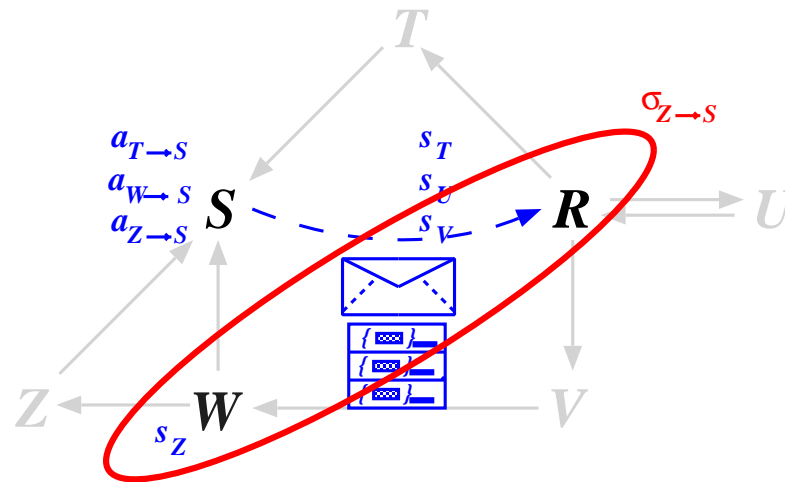


Privacy in the Face of Collusions



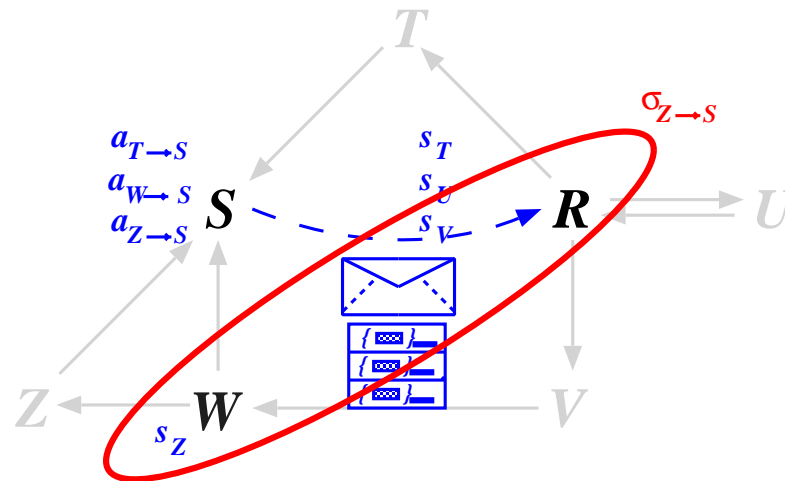
- Suppose R convinces W to help her check if $Z \stackrel{?}{\rightarrow} S$

Privacy in the Face of Collusions



- Suppose R convinces W to help her check if $Z \stackrel{?}{\rightarrow} S$
 - R can use the seed s_Z (revealed by W) to spot $\sigma_{Z \rightarrow S}$

Privacy in the Face of Collusions



- Suppose R convinces W to help her check if $Z \xrightarrow{?} S$
 - R can use the seed s_Z (revealed by W) to spot $\sigma_{Z \rightarrow S}$
- **Cause:** Backward authorization is **transferable**
- **Solution:** Non-transferability via **pairings**

Pairing Scheme Performance: Sender

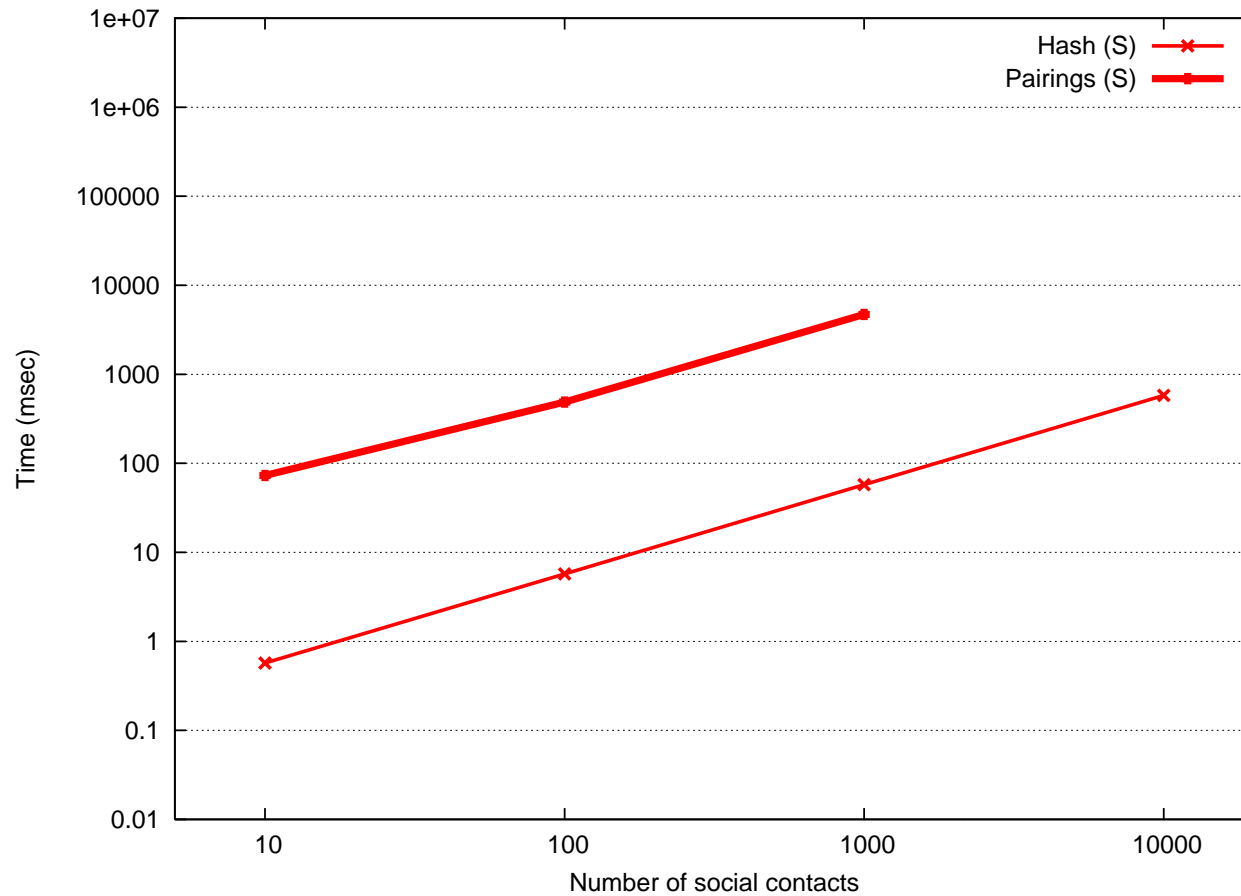


Table 2: Time (milliseconds) for proximity checks: pairing-based scheme.

Pairing Scheme Performance: Receiver

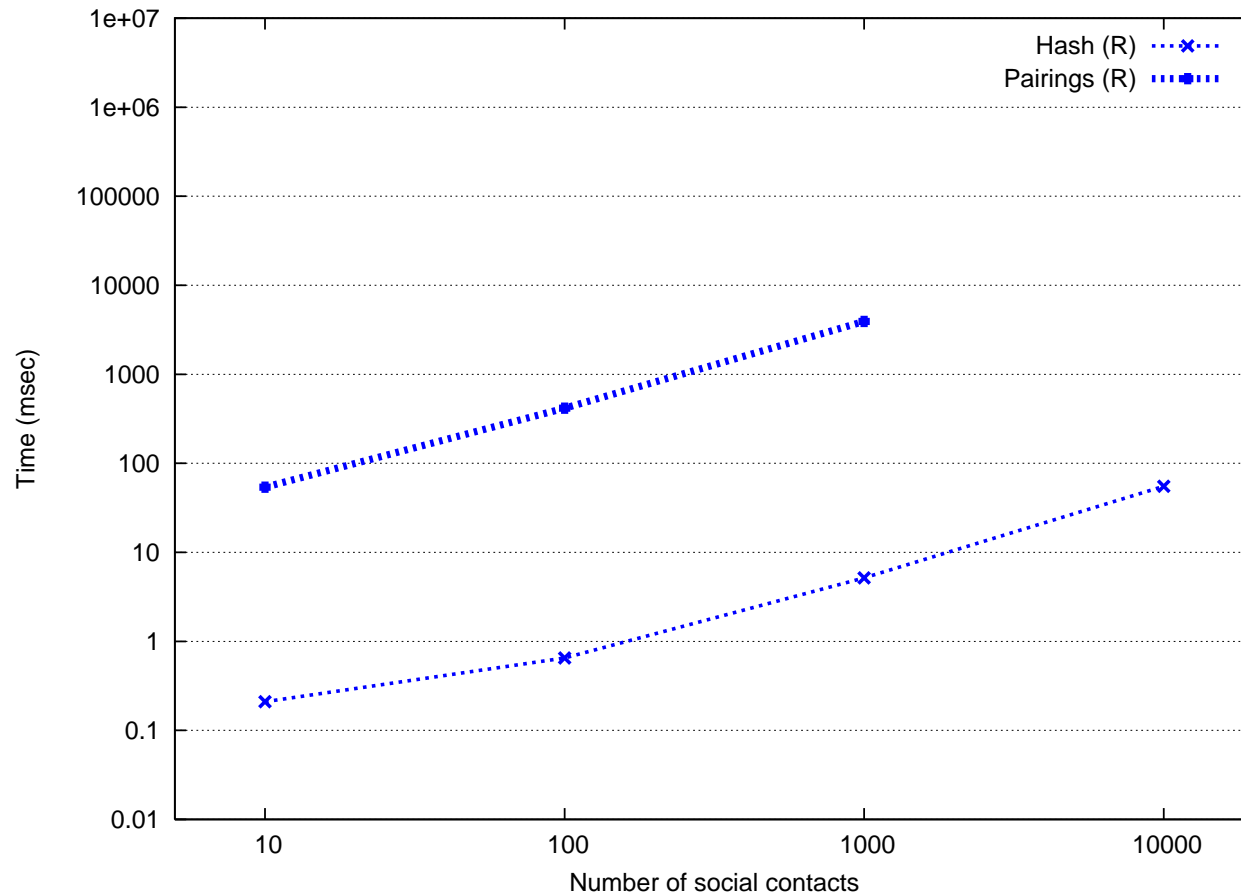


Table 2: Time (milliseconds) for proximity checks: pairing-based scheme.