Mylar

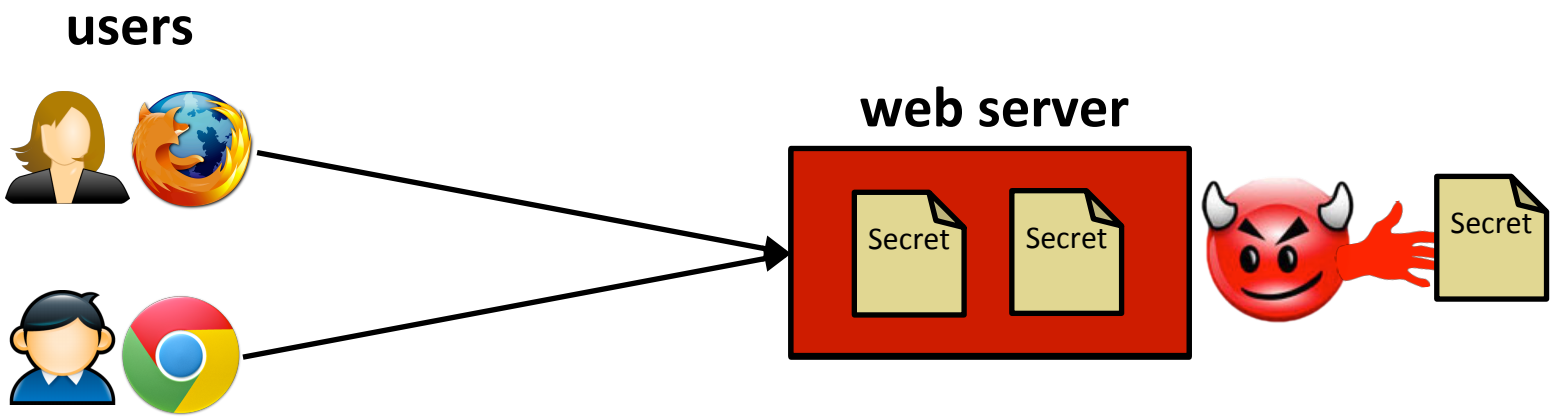
Building web applications on top of encrypted data

```
xd5d1db5abce2356d51db5aab23d5321535abbce23352abc4352314987
x435acb734352a12cad5d1db5abce2356d51db5345323acb2312aaab23
```

Raluca Ada Popa, Emily Stark, Jonas Helfer, Steven Valdez, Nickolai Zeldovich, M. Frans Kaashoek, and Hari Balakrishnan

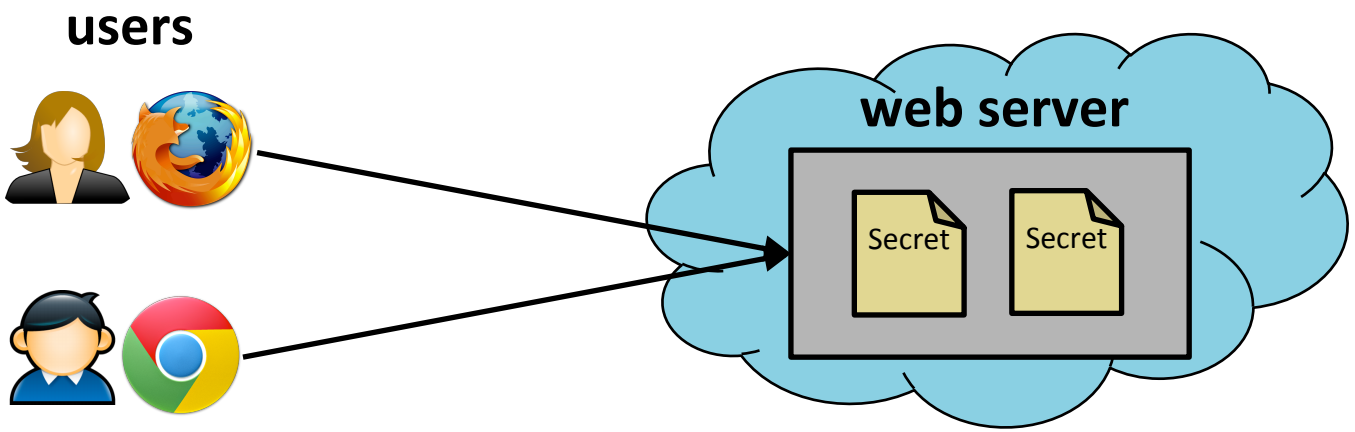
MIT CSAIL and Meteor Inc.

Problem



Confidential data leaks from web servers

Attackers get full server access



hackers

cloud employees

government

LivingSocial Hacked — More Than 50 Million Customer Names, Emails, Birthdates and Encrypted Passwor Accessed (Internal Memo)

APRIL 26, 2013 AT 1:15 PM PT

LivingSocial, the daily deals site

Why You Shouldn't Trust Facebook with Your Data: An Employee's Revelations

EXPAND

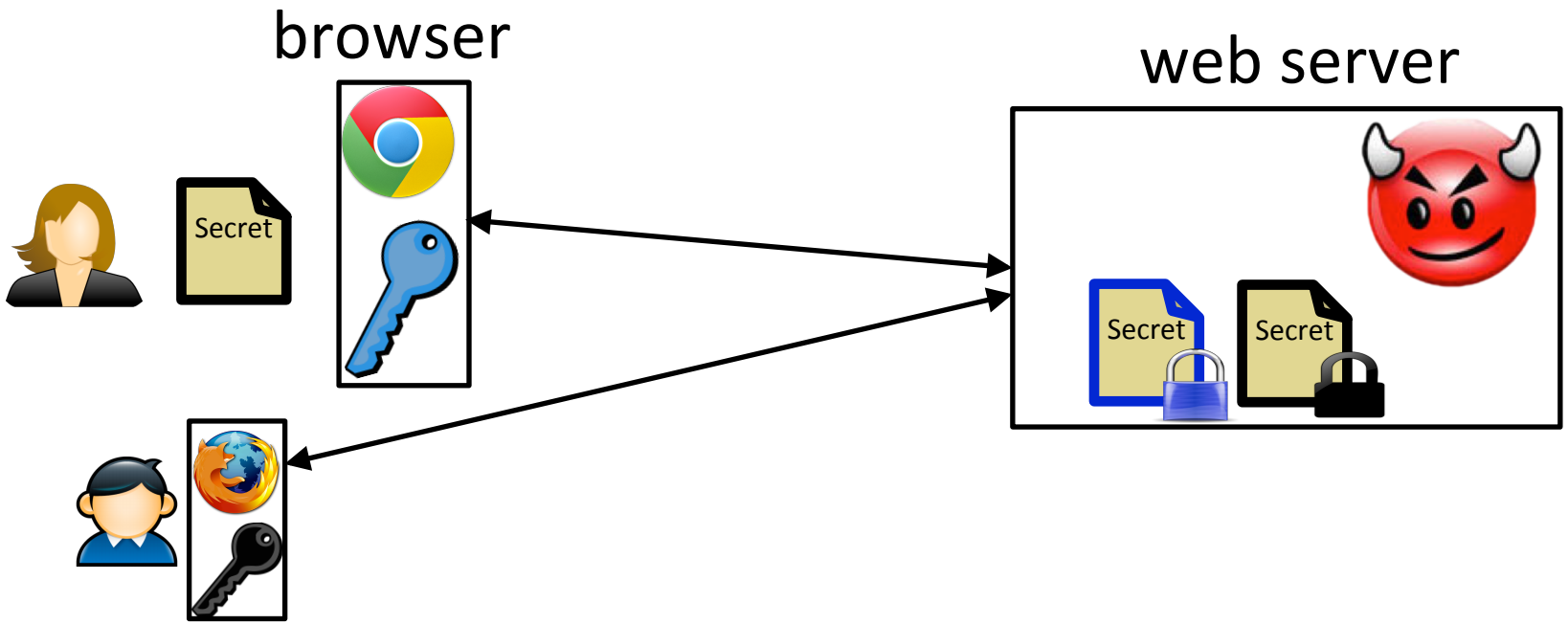
The abuse of private data by Facebook employees was pretty much inevitable;

Mylar

A web framework that protects confidentiality against fully compromised servers

Servers store data encrypted

Plaintext data exists only in browsers



Related work

- File systems: CFS, NCryptfs, SiRiUS, Plutus
- Encrypted databases: CryptDB, Monomi
- Browser encryption: Christodorescu'08, Cryptocat

Far from sufficient for real web apps

Challenges

- Active adversaries (e.g., corrupt webpage)
- Enabling functionality with encryption:
  - data sharing
  - computation

Mylar

- Active adversaries (e.g., corrupt webpage)
- Enabling functionality with encryption:
  - data sharing
  - computation

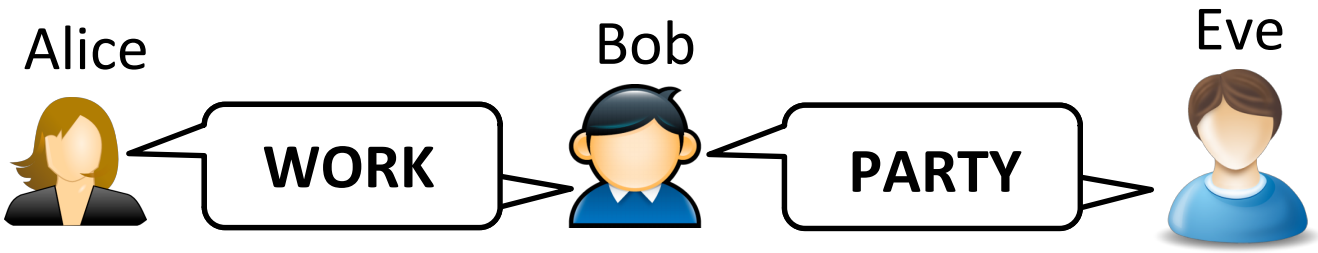
webpage code verification

client-side web framework

principal graph & certification

new encryption scheme:
multi-key search

Example: Chat application



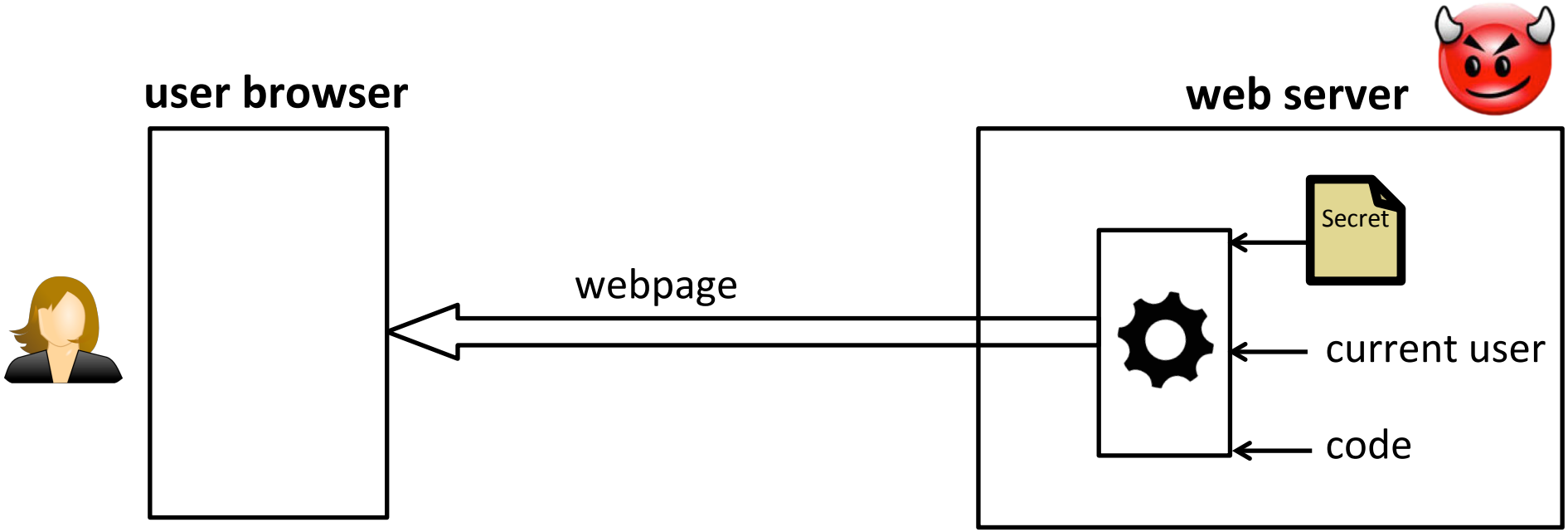
TODO:

- ☐ Server cannot see messages
- ☐ Users share chat rooms securely
- ☐ Format messages, generate html page
- ☐ Search

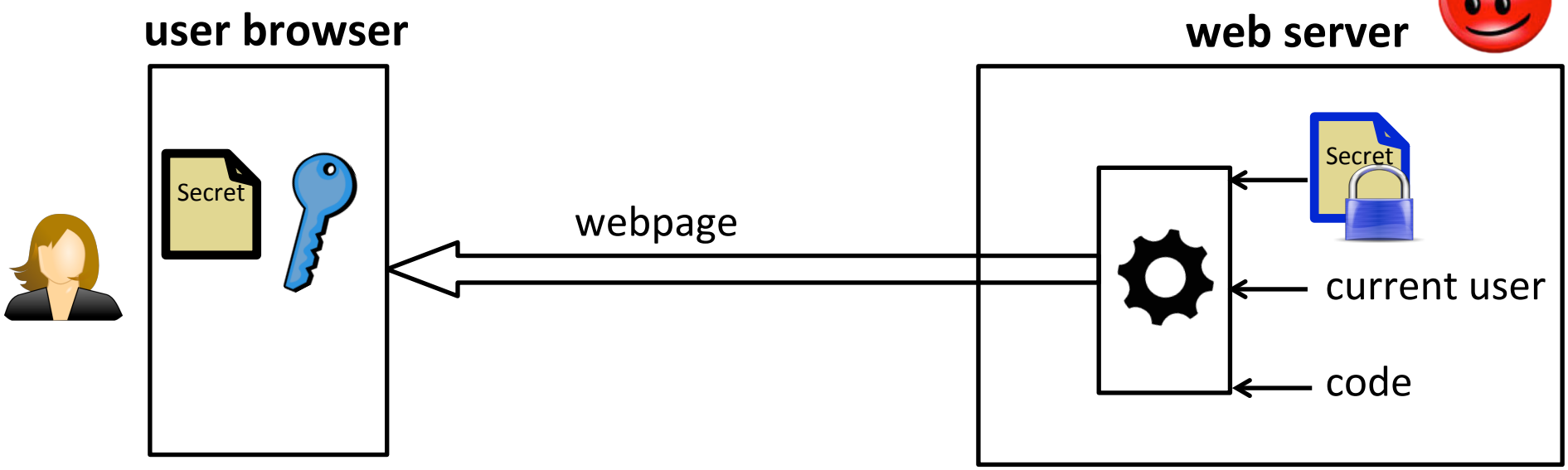
How to organize a web application framework for encryption?

Start: common web framework

e.g., Django, Ruby on Rails



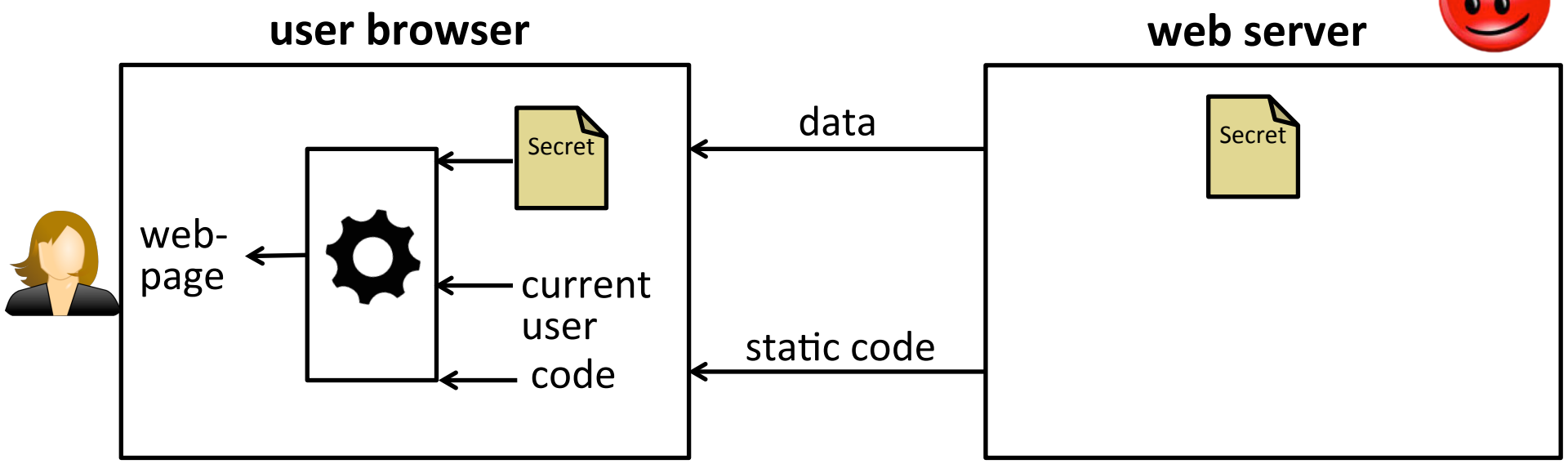
Add encryption



- server's computation is restricted by encryption
- easy to tamper with webpage

Client-side web framework

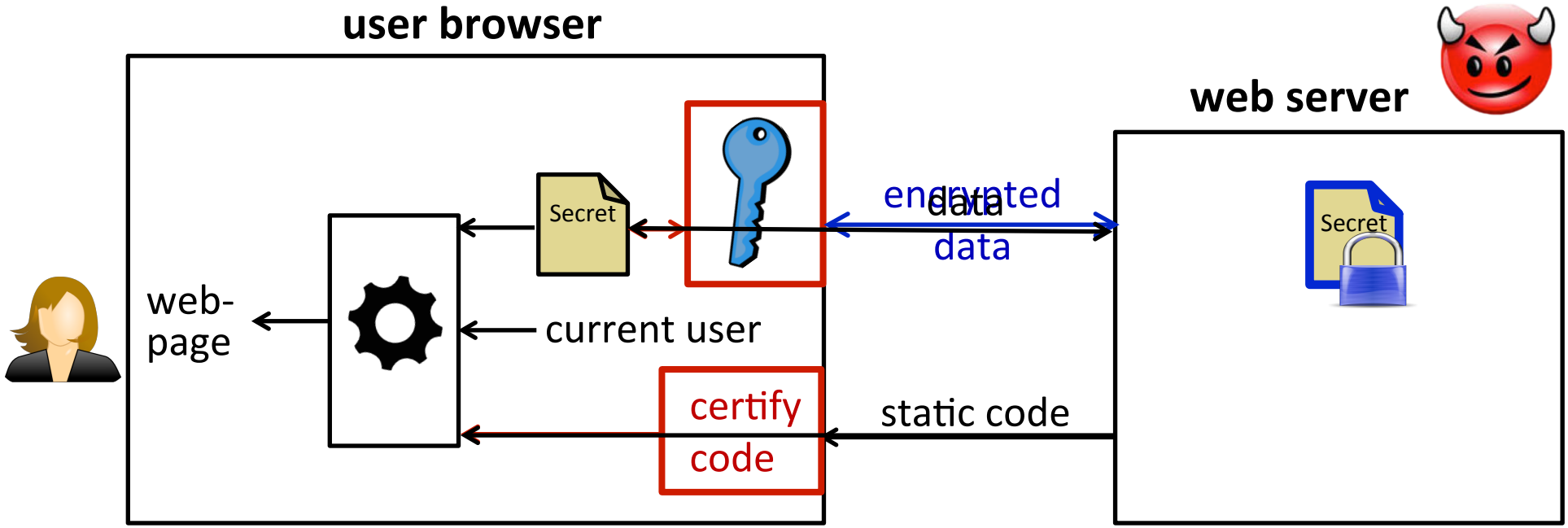
e.g., AJAX programming, Meteor



Data and code separate

Generate webpage at client, compute in browser

Mylar



Certify code (trusted developer)

Intercept and encrypt/decrypt data

Chat application


Server cannot see messages

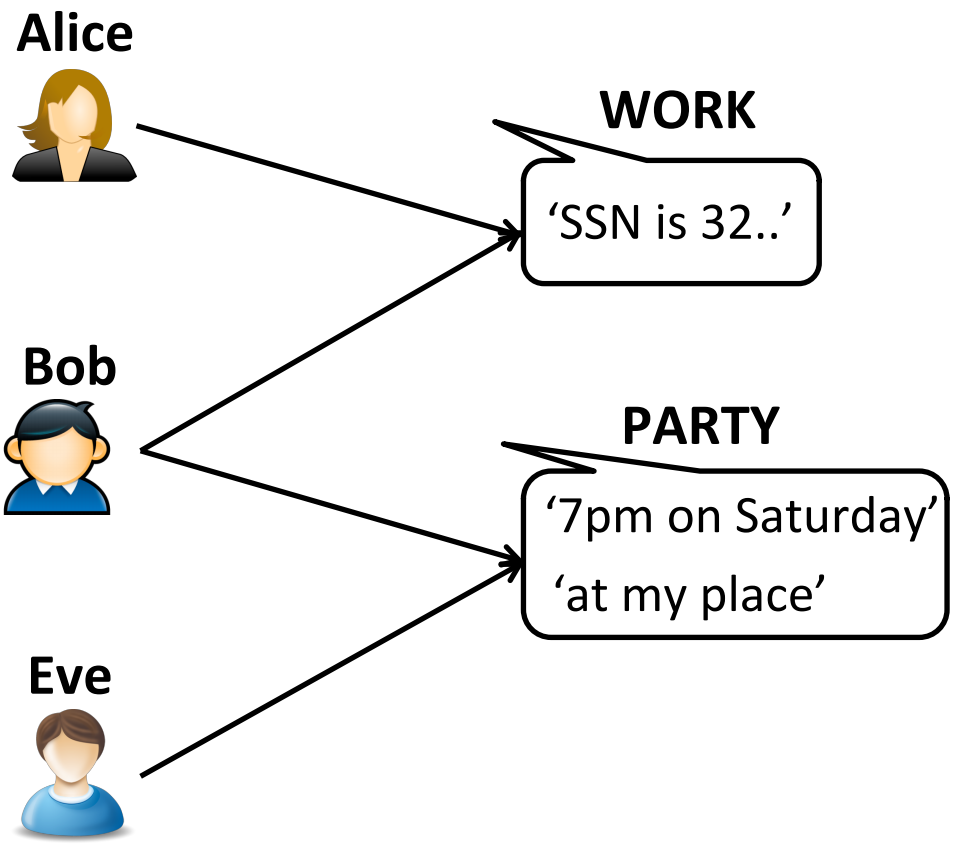

Format messages, generate html page


Users share chat rooms securely


Search

Data sharing

Developer specifies access control via the principal graph



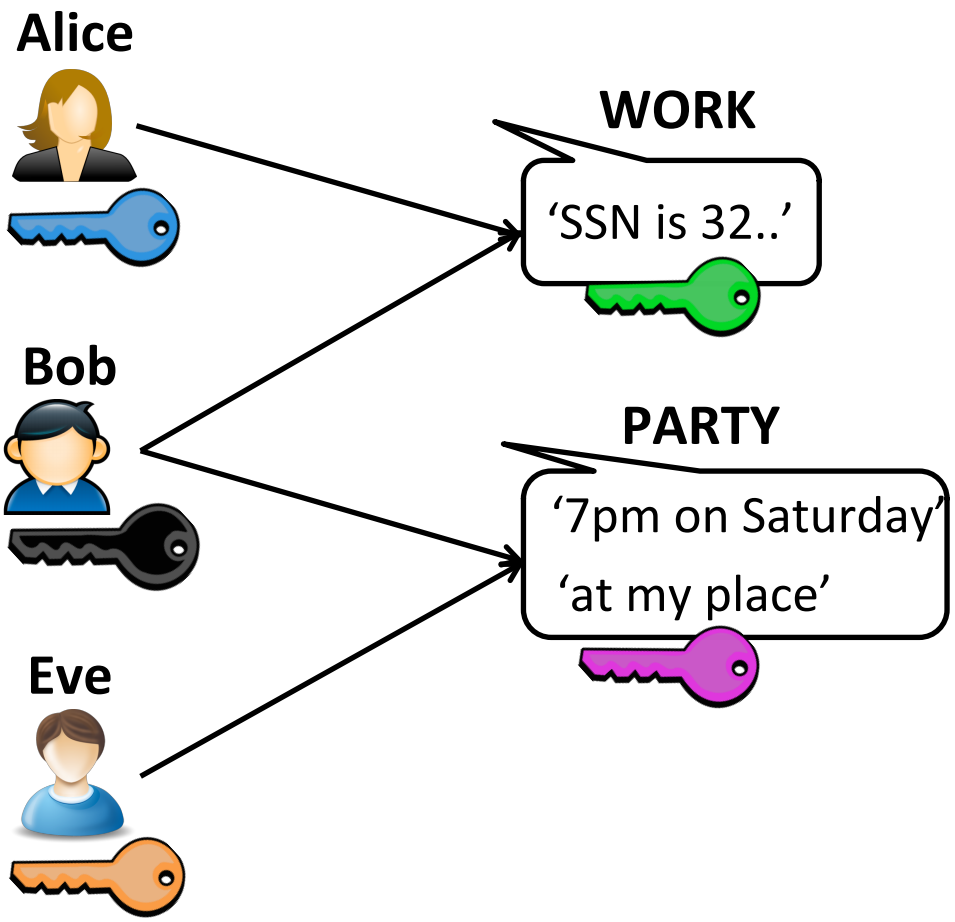
In Alice's browser:

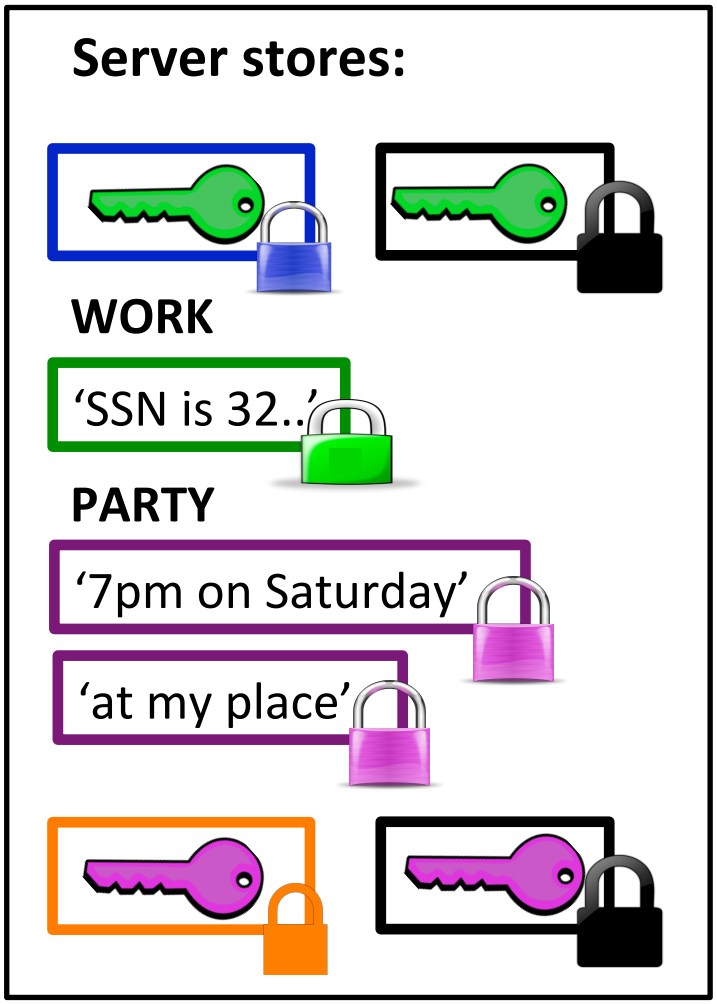
```
function create_chat(chattitle):
chat_princ = princ_create(chattitle,
                          princ_current());

function invite_user(username):
chat_princ.add_access(
                princ_lookup(username));

Messages.encrypted(
                {"message": chatprinc});

function send_message(msg):
Messages.insert({message: msg,
                 chat: cur_chat.id,
                 chatprinc: chat_princ});
```

Server database:

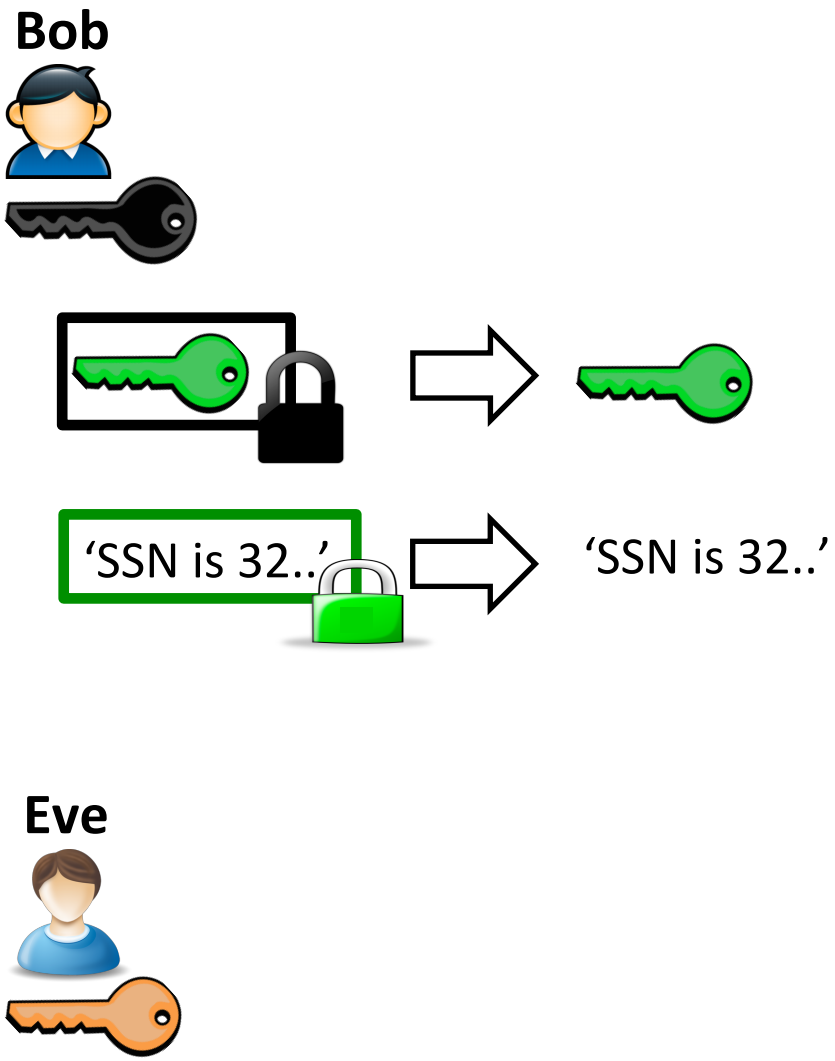
message	chat	chatprinc
'SSN is 32..'	WORK	WORK princ

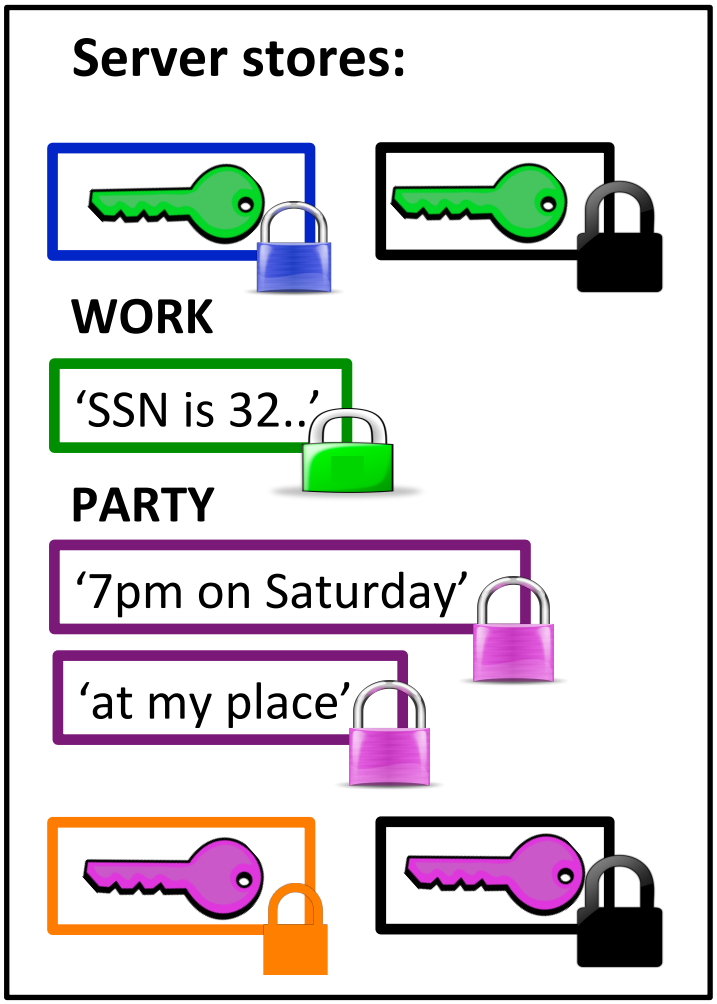
Enforce access with key chains



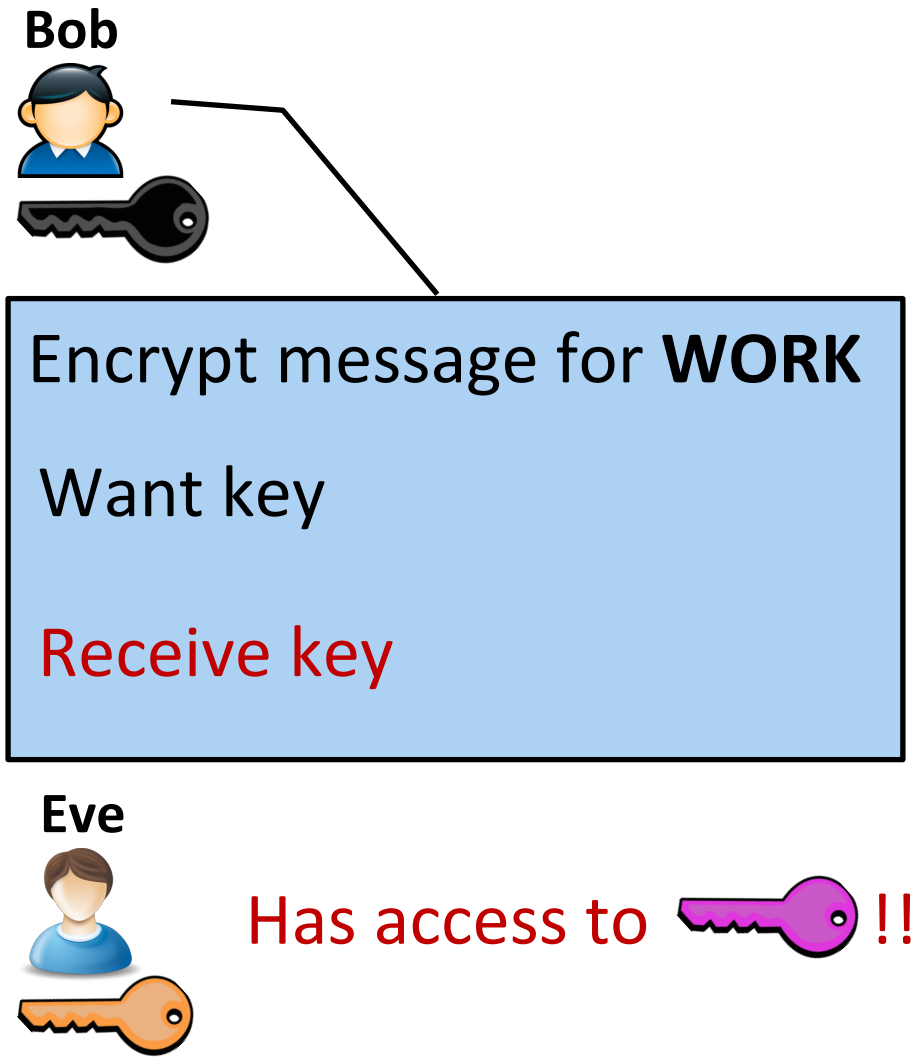


Get access to shared data

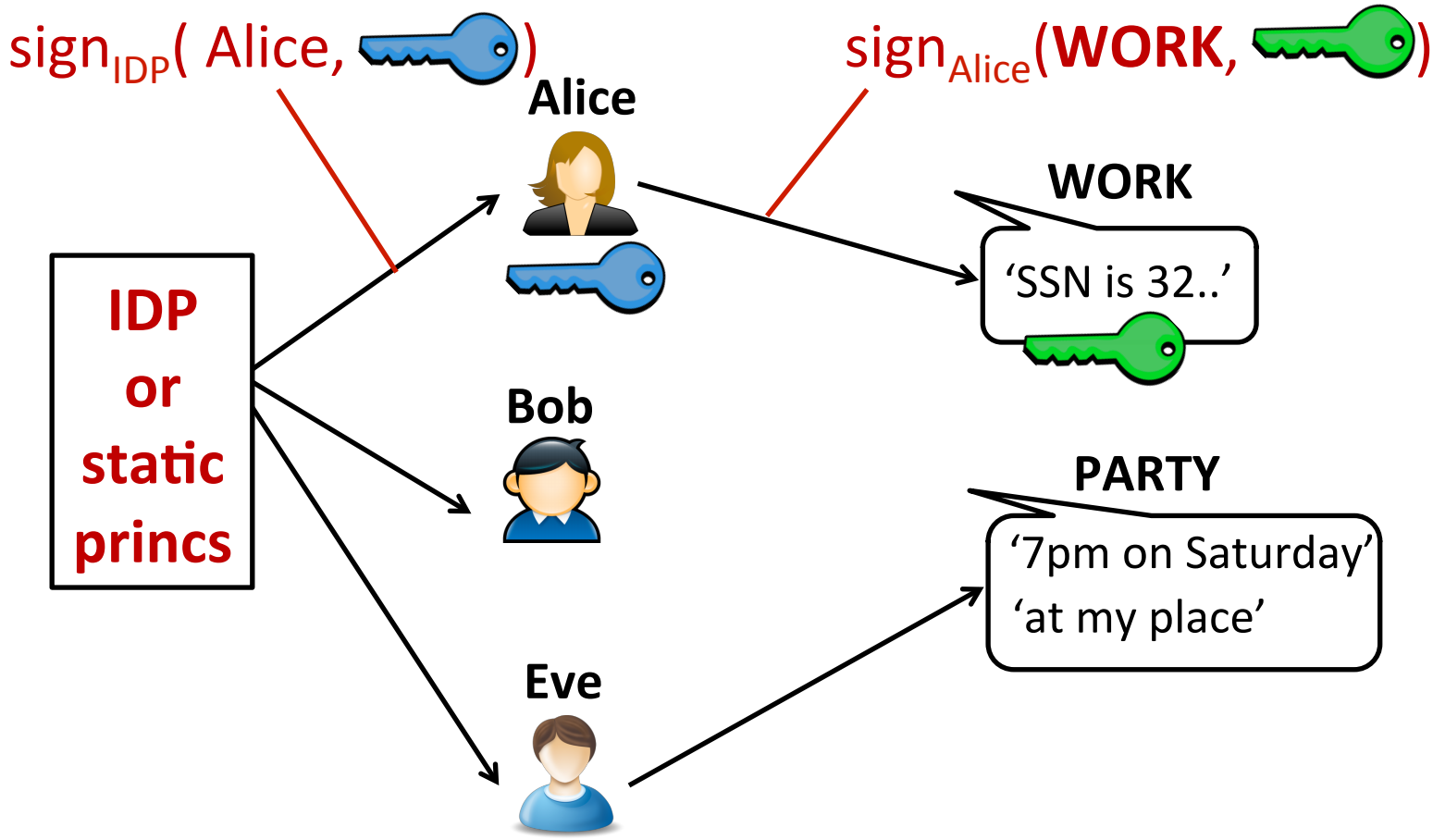




Problem: attacker gives incorrect key



Solution: Certification graph



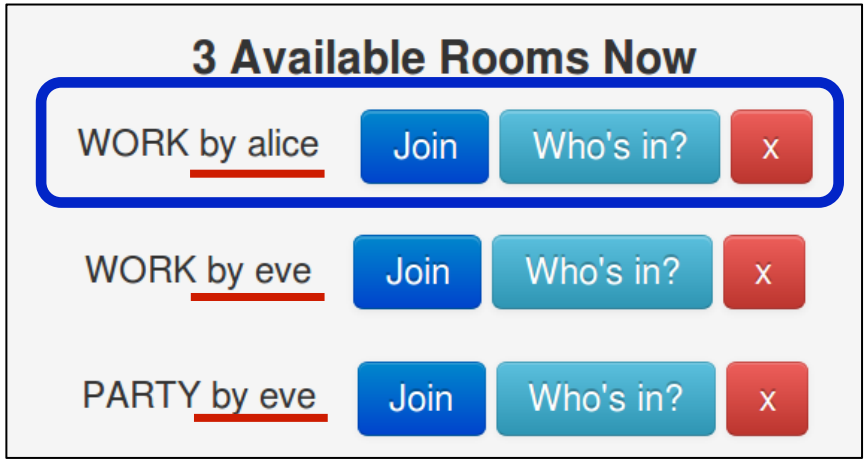
How does Bob's browser know

1. that it needs to check a signature from Alice?
2. Alice's PK? IDP: invoked once per user account creation

Choosing the certification path

1. Principals have human meaningful names
2. Developer displays entire path
3. User chooses path

with Mylar



without Mylar

No other change to user experience!

Chat application

Server cannot see messages

Format messages, generate html page

Users share chat rooms securely

Challenge: multi-key

Bob

Server:

WORK

‘SSN is 32..’

PARTY

‘7pm on Saturday’

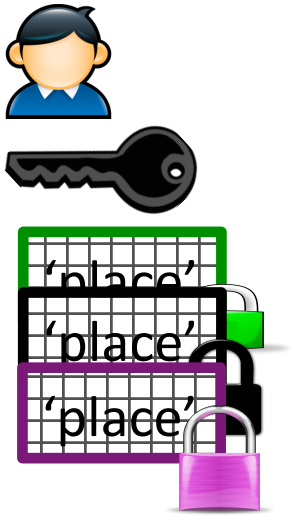
‘at my place’

Strawman: use single-key search scheme

[Kamara et al.'12]

Server:

Bob



WORK

'SSN is 32..'

PARTY

'7pm on Saturday'

'at my place'

Match!

Slow: pay overhead for each key

New cryptosystem: multi-key search

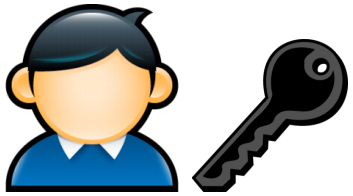
Server adjusts encryption from one key to another

Based on elliptic curves

API:

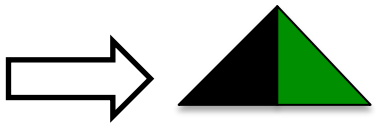
- Setup
- Keygen
- Encrypt
- Token
- Delta
- Adjust
- Match

Delta

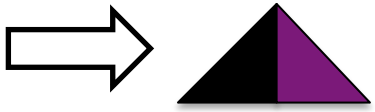


In Bob's browser:

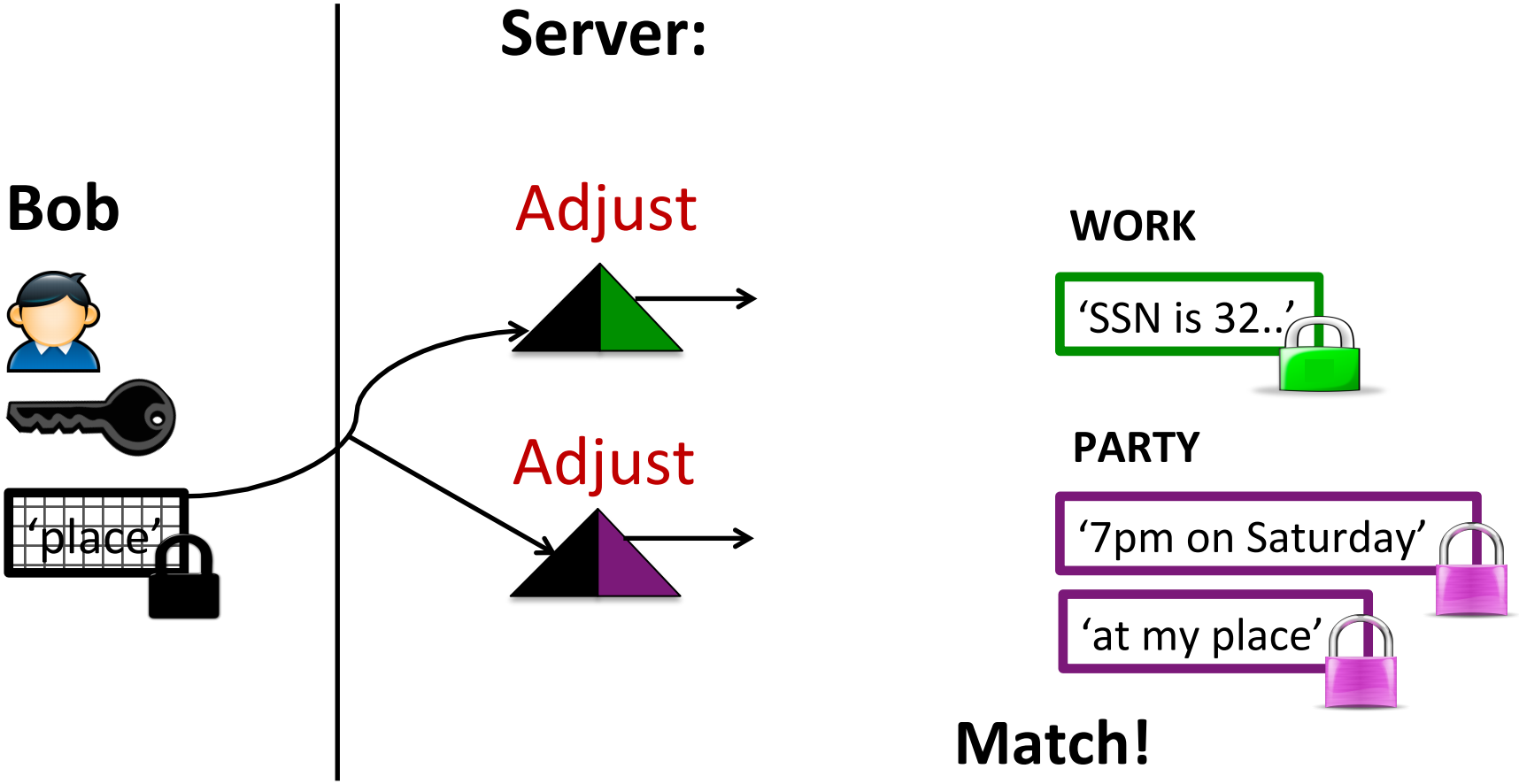
WORK:

Delta( , ) 

PARTY:

Delta( , ) 

Adjust


Server:
Bob
Adjust
Adjust
'place'
WORK
'SSN is 32..'
PARTY
'7pm on Saturday'
'at my place'
Match!

Chat application


Server cannot see messages


Format messages, generate html page


Users share chat rooms securely


Search

Confidentiality guarantees

Protects user A's data confidentiality against

- full server compromise
- compromise of any user machine, except for users with legitimate access to user A's data

assuming

- developer's client-side code does not leak data

Does not protect against side channels or access patterns, and does not hide metadata

Implementation

- On top of Meteor, but design is not limited to Meteor
- 9000 LoC: Javascript and C++

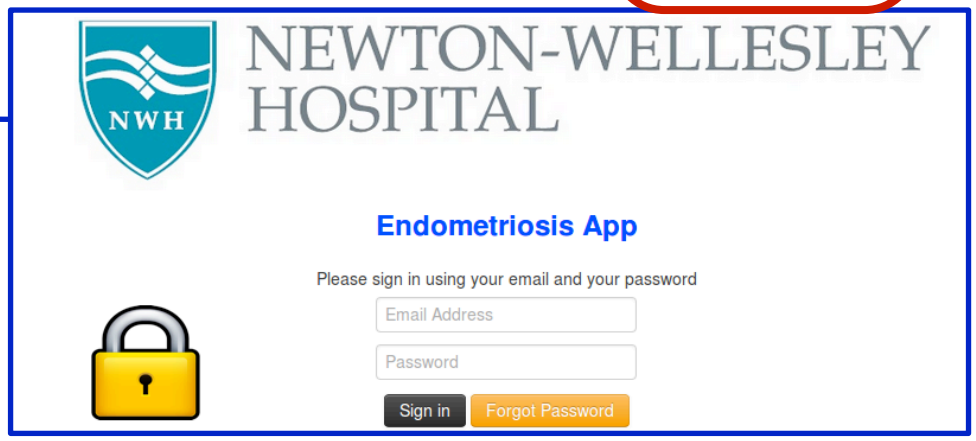
Evaluation

- How much developer effort does porting apps require?
- What is the performance overhead?

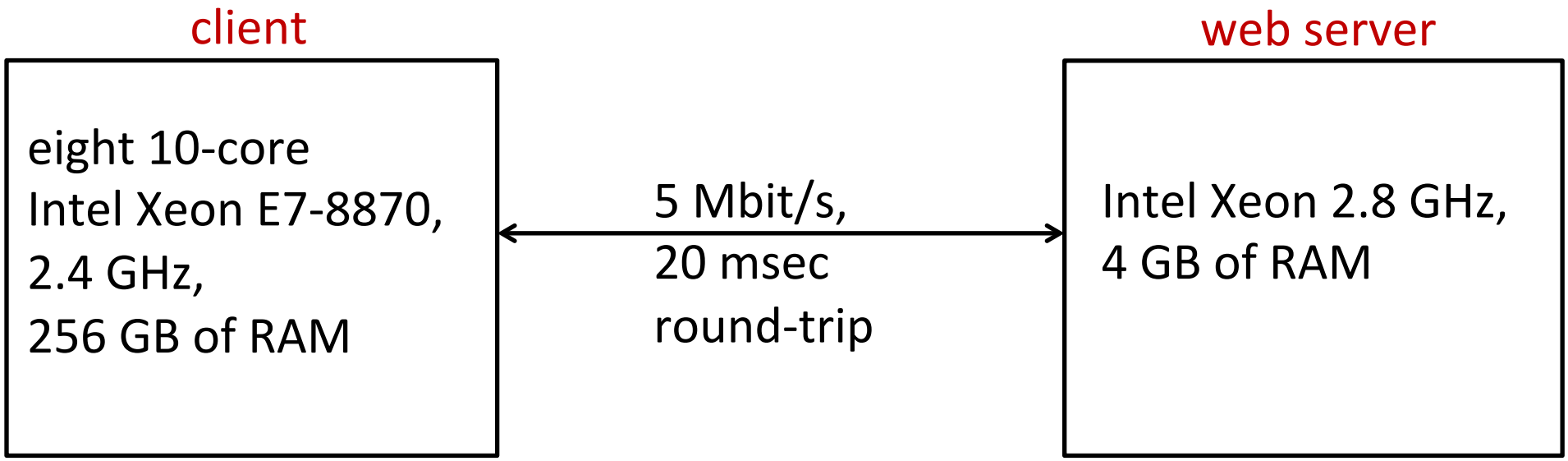
Applications

≈36 LoC

Applications	Fields secured	LoC added	LoC total	Existed before
kChat	chat messages	45	793	Yes
endometriosis	medical fields	28	3659	Yes
class submit	grades, homework, feedback	40	8410	Yes
photo sharing	photos, thumbnails, ..	32	610	No
forum	post body, title, ..	39	912	No
calendar	event body, title, …	30	798	No

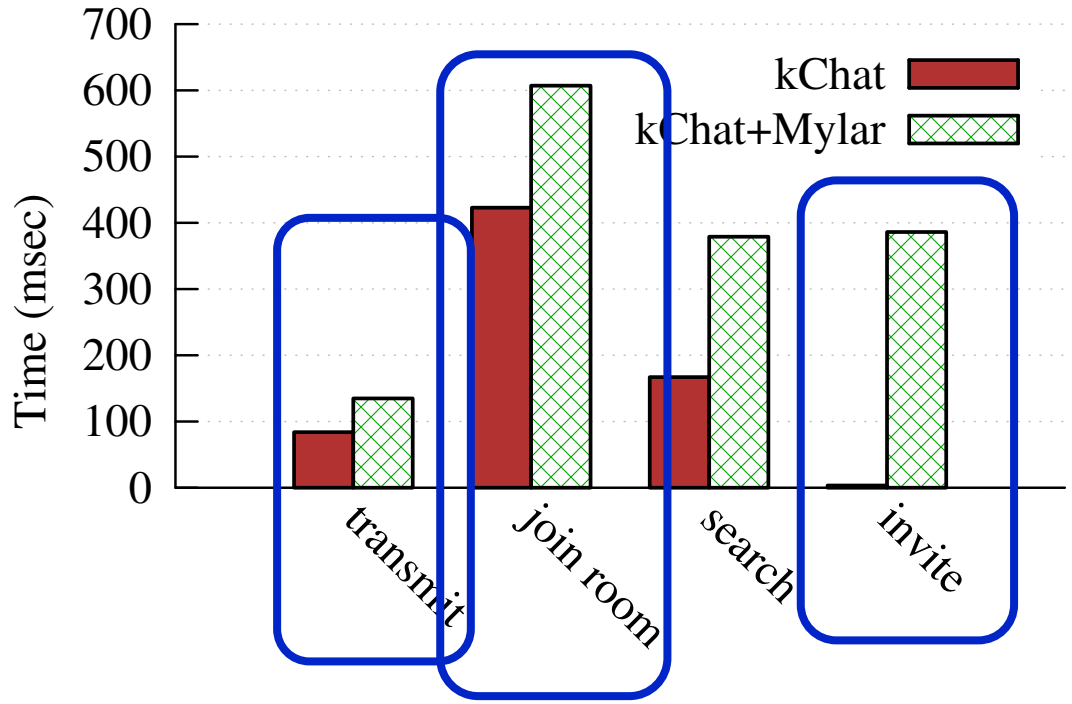


Experimental setup



kChat performance

Latency:



Throughput: 17% reduction

Mylar

- A web platform that protects confidentiality against full server compromise
  - Secures real applications with few LoC
  - Modest overhead

webpage code verification

principal graph & certification

new encryption scheme: multi-key search

http://css.csail.mit.edu/mylar/

Demo!