Privacy, Discovery, and Authentication for the Internet of Things

David Wu

Joint work with Ankur Taly, Asim Shankar, and Dan Boneh

The Internet of Things (IoT)

Security and Alarm
Light Control
Motion Sensor
HVAC Control
Door Control
Environmental Control
Window Control
STB
Remote Control

Lots of smart devices, but only useful if users can discover them!

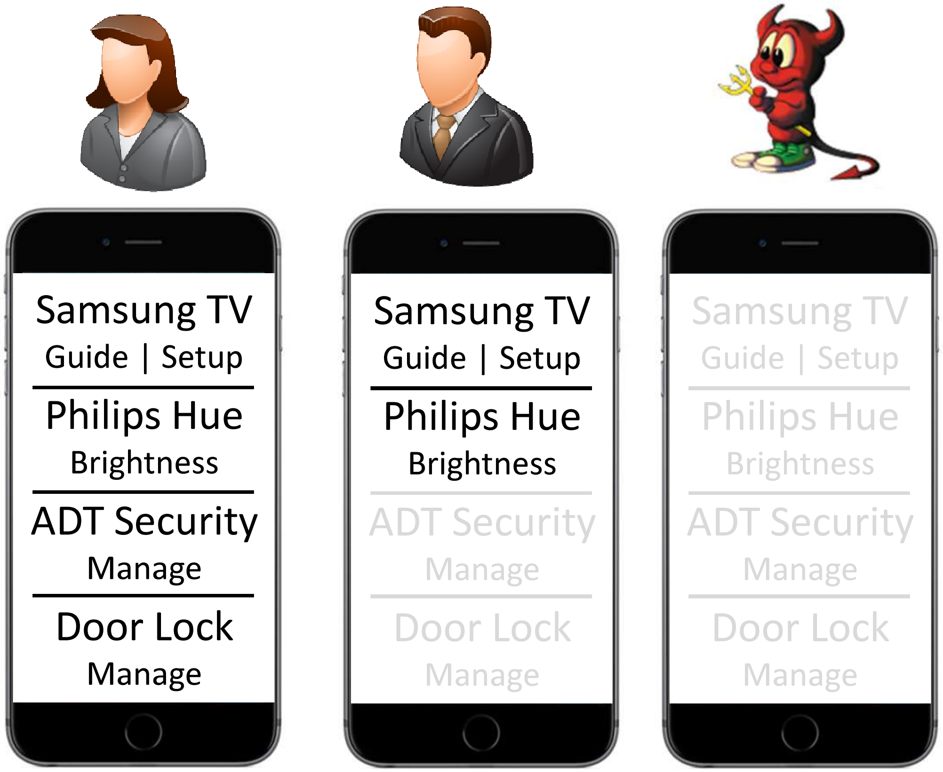
Private Service Discovery

- Many existing service discovery protocols: Multicast DNS (mDNS), Apple Bonjour, Bluetooth Low Energy (BLE)
- But… not much privacy
  - Recent study of mDNS announcements by Könings et al. [KBSW13] show that nearly 60% of devices revealed the device owner’s name in the clear (across approximately 3000 devices on a university campus)

- Service advertisements are not authenticated: malicious devices can forge service broadcasts

Private Service Discovery

Each service specifies an authorization policy



Samsung TV
Guide | Setup
Philips Hue
Brightness
ADT Security
Manage
Door Lock
Manage

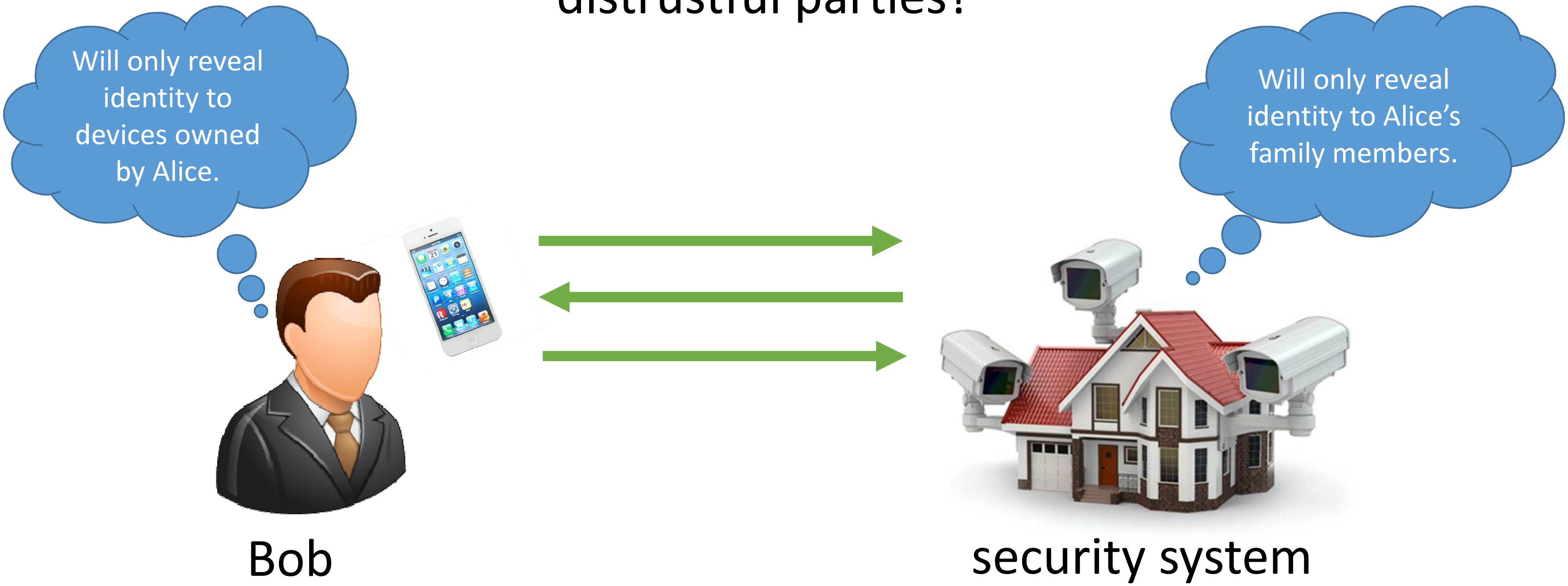
Samsung TV
Guide | Setup
Philips Hue
Brightness
ADT Security
Manage
Door Lock
Manage

Samsung TV
Guide | Setup
Philips Hue
Brightness
ADT Security
Manage
Door Lock
Manage

Alice Guest Stranger

Private Mutual Authentication

How to authenticate between mutually distrustful parties?



Will only reveal identity to devices owned by Alice.

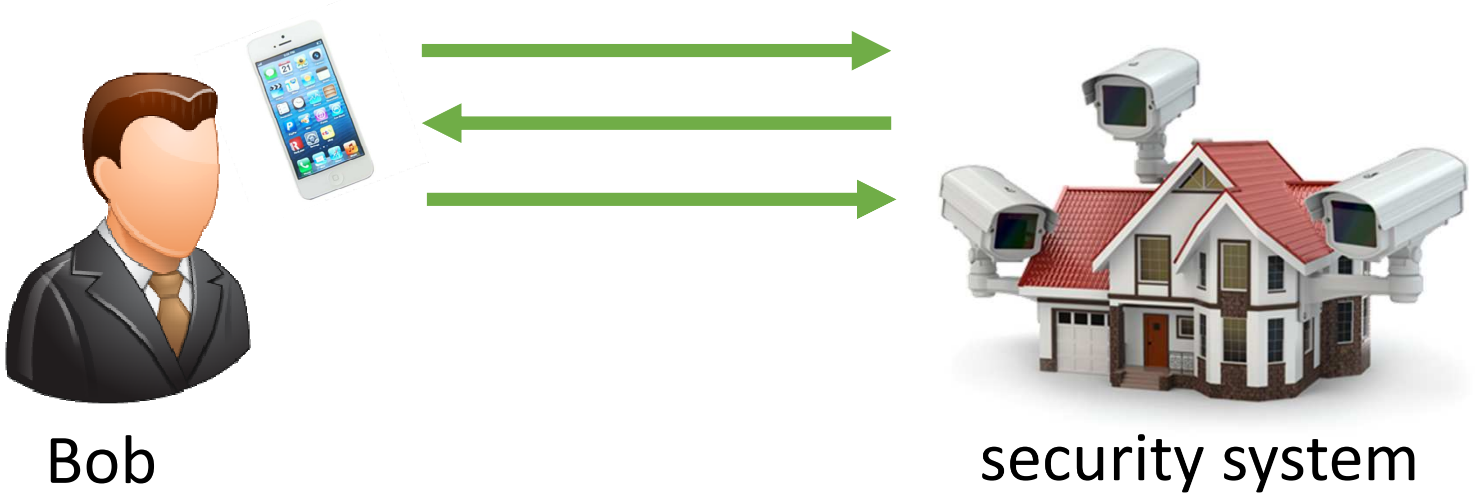
Will only reveal identity to Alice's family members.

Bob

security system

Private Mutual Authentication

In most existing mutual authentication protocols (e.g., TLS, IKE, SIGMA), one party must reveal its identity first



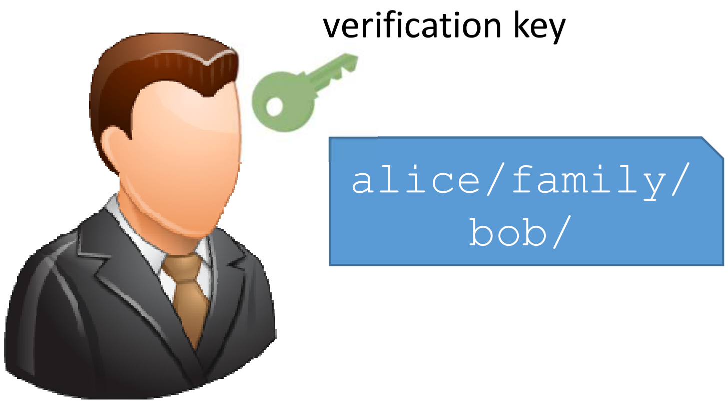
Primary Protocol Requirements

- **Mutual privacy:** Identity of protocol participants are only revealed to authorized recipients

- **Authentic advertisements:** Service advertisements (for discovery) should be unforgeable and authentic

Identity and Authorization Model

Every party has a signing + verification key, and a collection of human-readable names bound to their public keys via a certificate chain



verification key

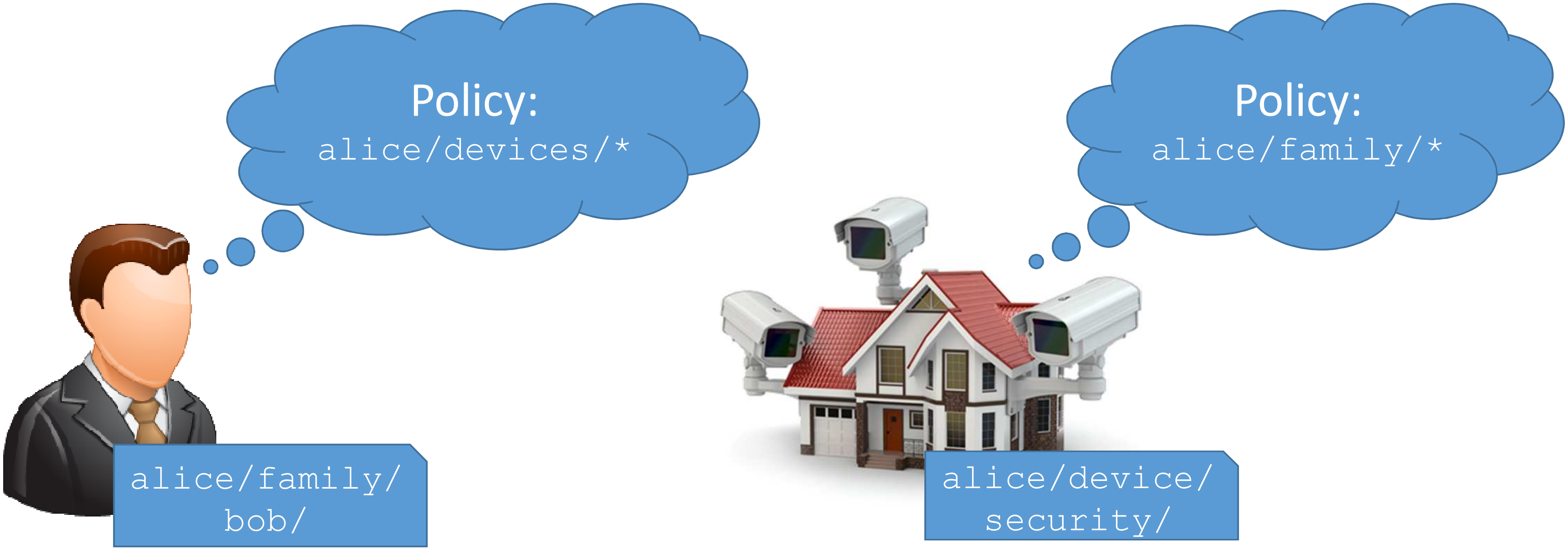
alice/family/bob/

alice/device/security/

popular_corp/prod/S1234

Identity and Authorization Model

Authorization decisions expressed as prefix patterns



Policy:

`alice/devices/*`

`alice/family/bob/`

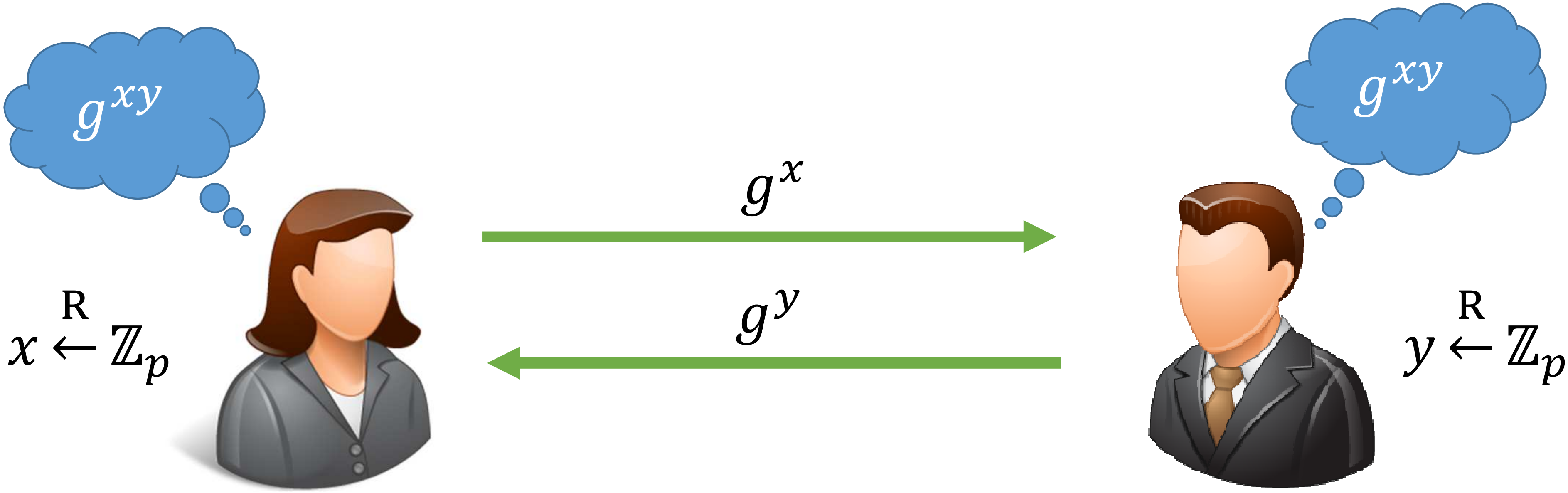
Policy:

`alice/family/*`

`alice/device/security/`

Protocol Construction

Starting Point: Diffie-Hellman Key Exchange



g^{xy}

$x \overset{\text{R}}{\leftarrow} \mathbb{Z}_p$

g^x

g^y

g^{xy}

$y \overset{\text{R}}{\leftarrow} \mathbb{Z}_p$

$\mathbb{G}$: cyclic group of prime order p
with generator g

Shared key:
$\mathrm{KDF}(g^x, g^y, g^{xy})$

Starting Point: Diffie-Hellman Key Exchange

$x \overset{R}{\leftarrow} \mathbb{Z}_p$

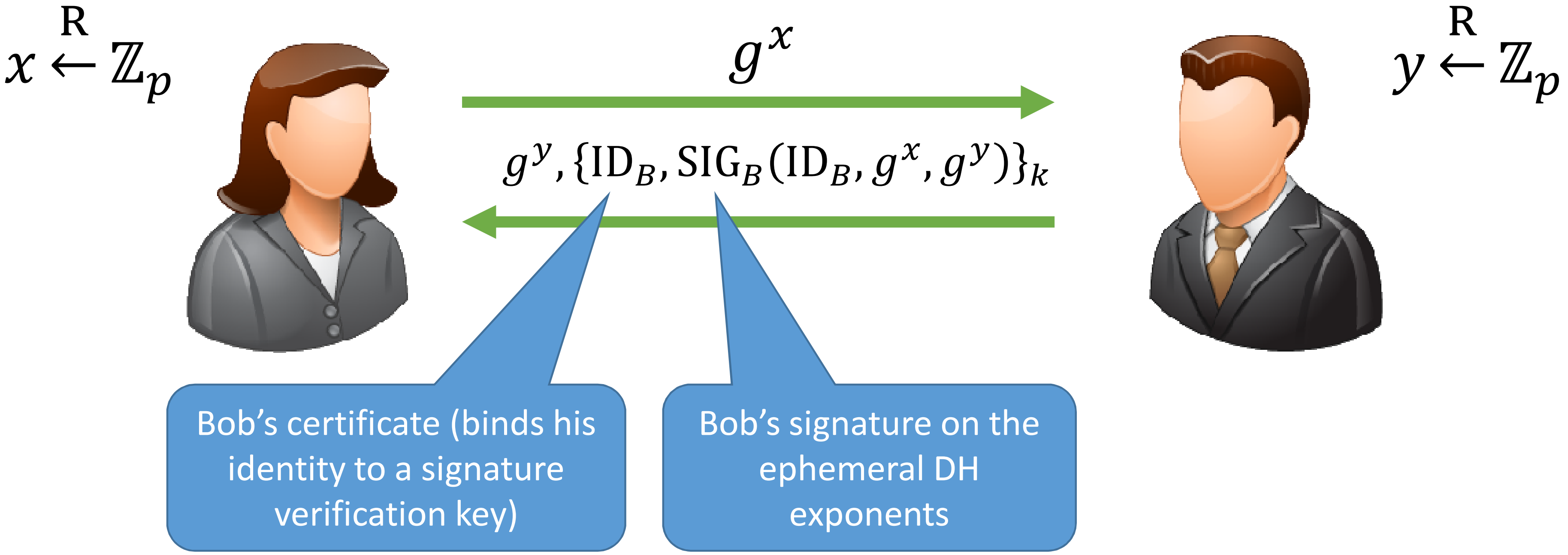
$y \overset{R}{\leftarrow} \mathbb{Z}_p$

NOT SECURE!

group of prime order p
with generator g

Shared key:
$\mathrm{KDF}(g^x, g^y, g^{xy})$

Secure Key Agreement: SIGMA-I Protocol [CK01]



$x \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_p$

$y \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_p$

g^x

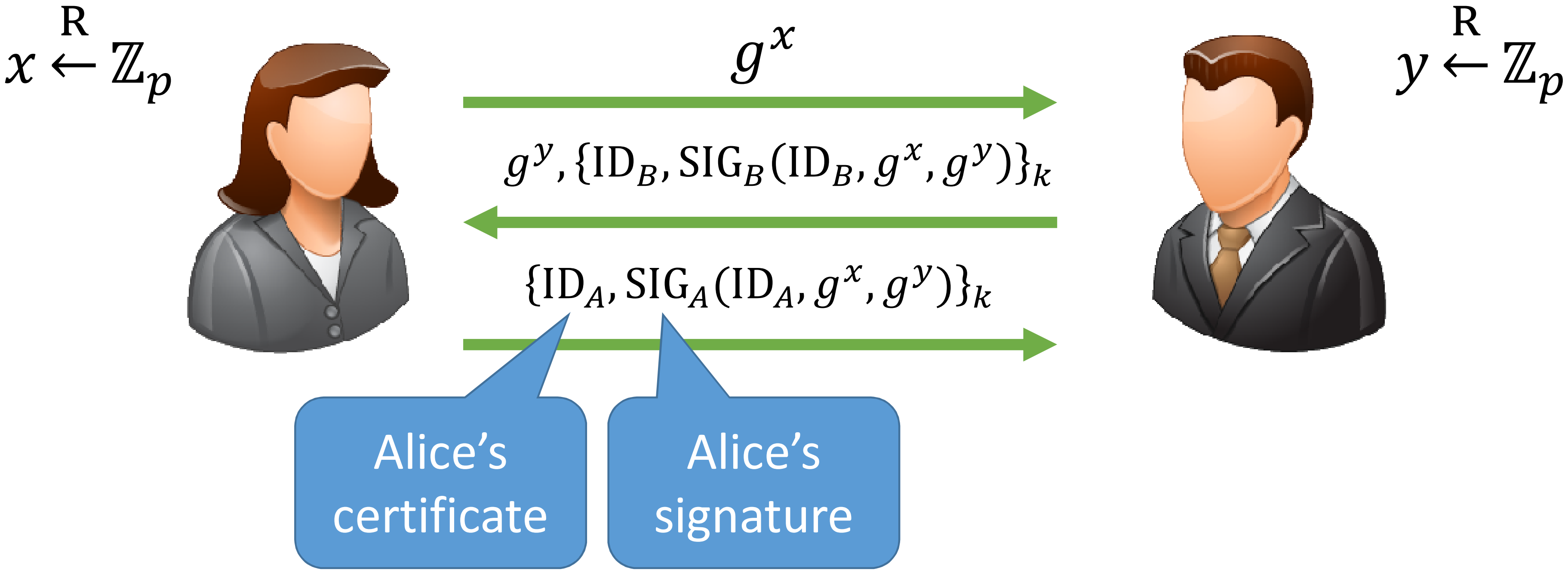
$g^y, \{\mathrm{ID}_B, \mathrm{SIG}_B(\mathrm{ID}_B, g^x, g^y)\}_k$

Bob’s certificate (binds his identity to a signature verification key)

Bob’s signature on the ephemeral DH exponents

Key requirement: some form of authentication

Secure Key Agreement: SIGMA-I Protocol [CK01]



$x \overset{\text{R}}{\leftarrow} \mathbb{Z}_p$

$y \overset{\text{R}}{\leftarrow} \mathbb{Z}_p$

g^x

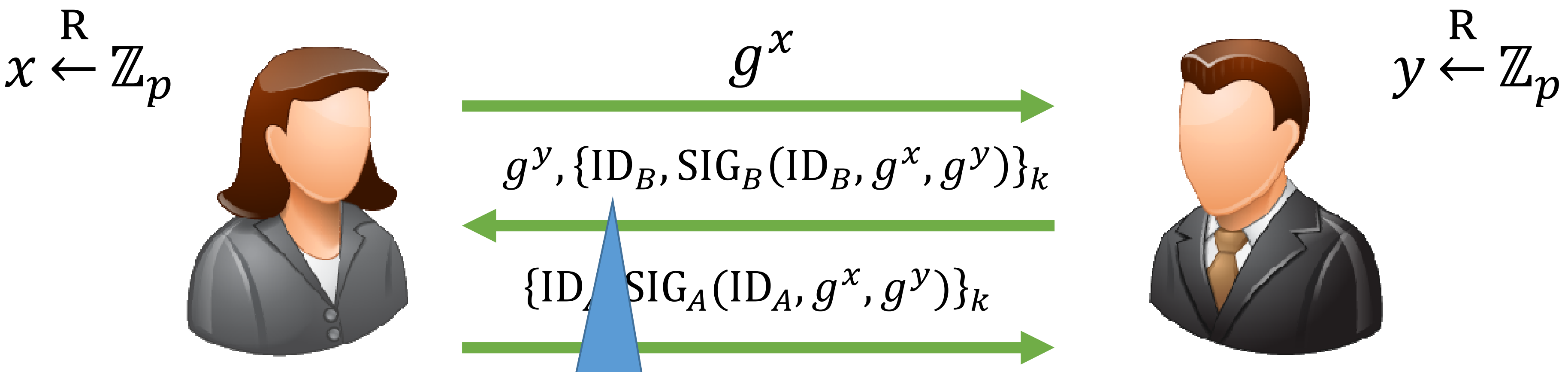
$g^y, \{\text{ID}_B, \text{SIG}_B(\text{ID}_B, g^x, g^y)\}_k$

$\{\text{ID}_A, \text{SIG}_A(\text{ID}_A, g^x, g^y)\}_k$

Alice’s certificate

Alice’s signature

Secure Key Agreement: SIGMA-I Protocol [CK01]



$x \xleftarrow{R} \mathbb{Z}_p$

$y \xleftarrow{R} \mathbb{Z}_p$

g^x

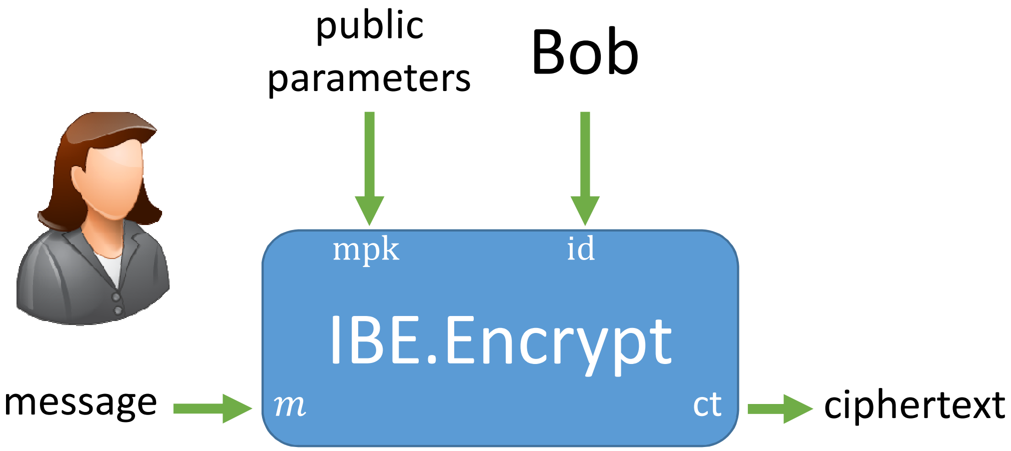
$g^y, \{\mathrm{ID}_B, \mathrm{SIG}_B(\mathrm{ID}_B, g^x, g^y)\}_k$

$\{\mathrm{ID}_A, \mathrm{SIG}_A(\mathrm{ID}_A, g^x, g^y)\}_k$

Bob sends his identity before learning anything about Alice's identity!

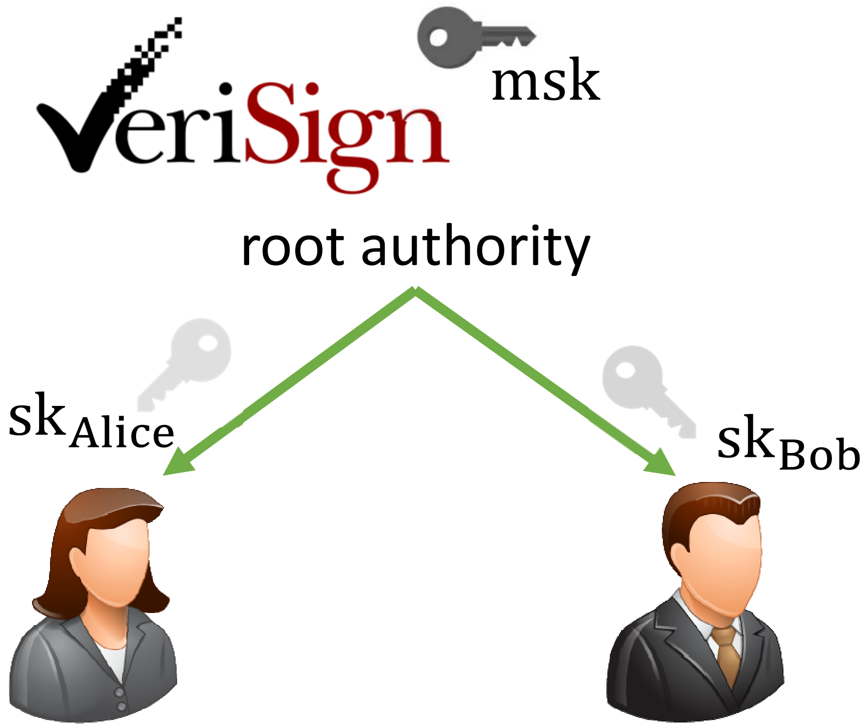
Identity Based Encryption (IBE) [Sha84, BF01, Coc01]

Public-key encryption scheme where public-keys can be arbitrary strings (identities)



Alice can encrypt a message to Bob without needing to have exchanged keys with Bob

Identity Based Encryption (IBE) [Sha84, BF01, Coc01]

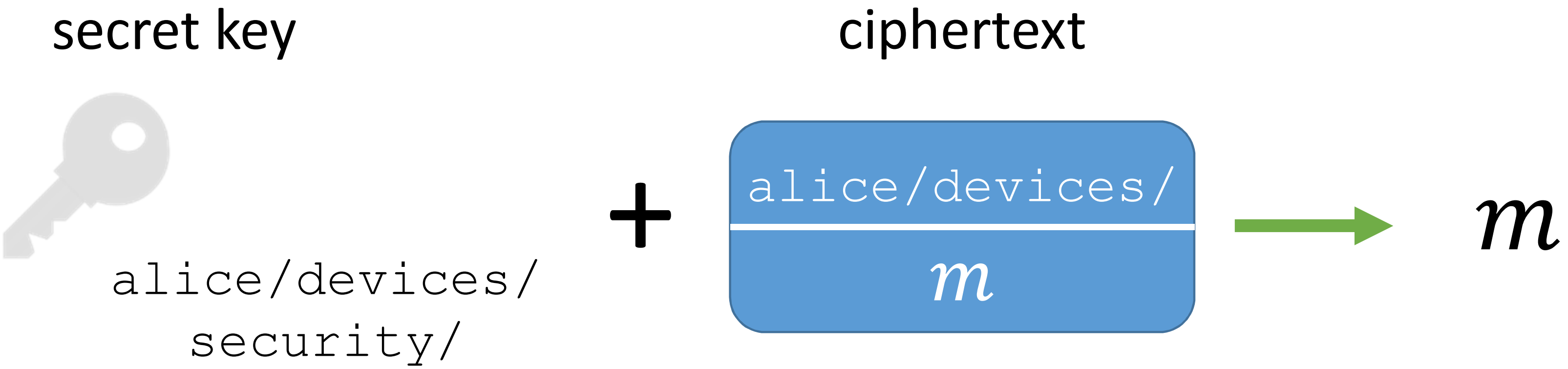


To decrypt messages, users go to a (trusted) identity provider to obtain a decryption key for their identity

Bob can decrypt all messages encrypted to his identity using $\mathrm{sk}_{\mathrm{Bob}}$

Prefix-Based Encryption

Secret-keys and ciphertexts both associated with names



secret key

alice/devices/
security/

+

ciphertext

alice/devices/

m

m

Decryption succeeds if name in ciphertext is a prefix of the name in the secret key

Prefix-Based Encryption

Can be leveraged for prefix-based policies

Policy:
alice/devices/*

Bob encrypts his message to the identity `alice/devices/`. Any user with a key that begins with `alice/devices/` can decrypt.

Prefix-Based Encryption

Can be leveraged for prefix-based policies

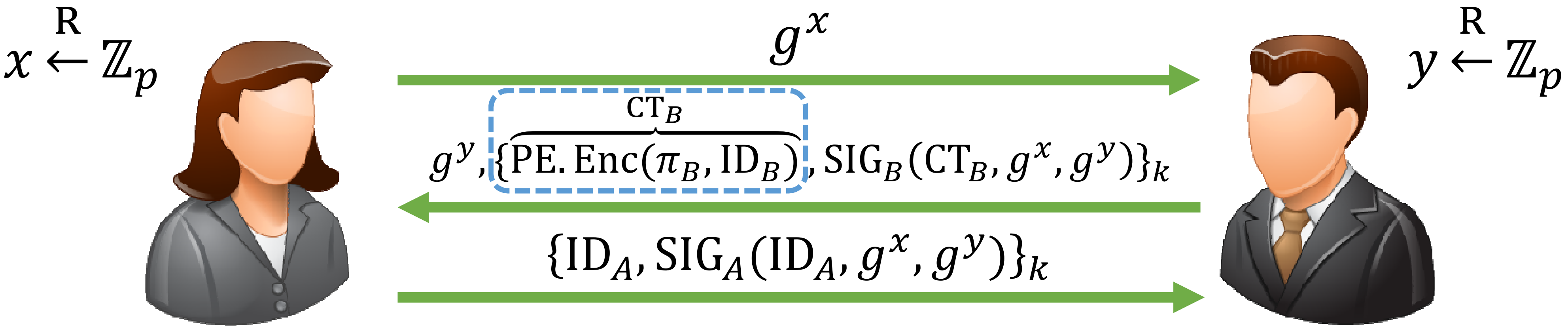
Policy:
`alice/devices/*`

Bob en the
identit Any
user w ith
`alice/` crypt.

Can be built directly from IBE!

Private Mutual Authentication

Key idea: encrypt certificate using prefix-based encryption



$x \xleftarrow{\mathrm{R}} \mathbb{Z}_p$

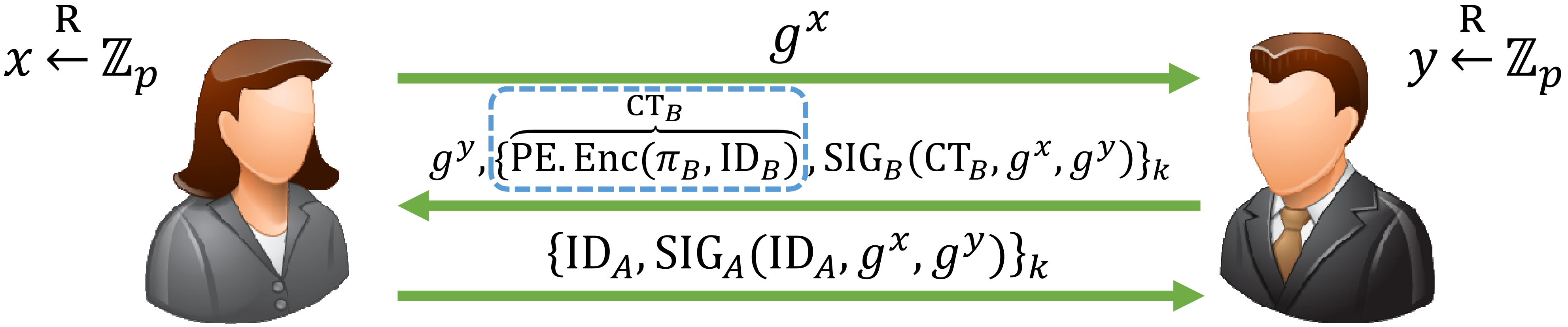
g^x

$g^y, \{\overbrace{\mathrm{PE.Enc}(\pi_B, \mathrm{ID}_B)}^{\mathrm{CT}_B}, \mathrm{SIG}_B(\mathrm{CT}_B, g^x, g^y)\}_k$

$\{\mathrm{ID}_A, \mathrm{SIG}_A(\mathrm{ID}_A, g^x, g^y)\}_k$

$y \xleftarrow{\mathrm{R}} \mathbb{Z}_p$

Private Mutual Authentication



$x \xleftarrow{\mathrm{R}} \mathbb{Z}_p$

g^x

$g^y, \boxed{\overbrace{\{\mathrm{PE.Enc}(\pi_B, \mathrm{ID}_B)}^{\mathrm{CT}_B}}, \mathrm{SIG}_B(\mathrm{CT}_B, g^x, g^y)\}_k$

$\{\mathrm{ID}_A, \mathrm{SIG}_A(\mathrm{ID}_A, g^x, g^y)\}_k$

$y \xleftarrow{\mathrm{R}} \mathbb{Z}_p$

- **Privacy for Alice’s identity:** Alice sends her identity only after verifying Bob’s identity
- **Privacy for Bob’s identity:** Only users with a key that satisfies Bob’s policy can decrypt his identity

Private Service Discovery

Prefix-based encryption can also be leveraged for *private* service discovery

See paper for details:
http://arxiv.org/abs/1604.06959

Implementation and Benchmarks

- Integrated private mutual authentication and private service discovery protocols into the Vanadium open-source framework for building distributed applications

https://github.com/vanadium/

Implementation and Benchmarks

	Intel Edison	Raspberry Pi	Nexus 5X	Laptop	Desktop
SIGMA-I	252.1 ms	88.0 ms	91.6 ms	6.3 ms	5.3 ms
Private Mutual Auth.	1694.3 ms	326.1 ms	360.4 ms	19.6 ms	9.5 ms
Slowdown	6.7x	3.7x	3.9x	3.1x	1.8x

Comparison of private mutual authentication protocol with non-private SIGMA-I protocol

Note: x86 assembly optimizations for pairing curve operations available only on desktop

Conclusions

- Existing key-exchange and service discovery protocols do not provide privacy controls

- Prefix-based encryption can be combined very naturally with existing key-exchange protocols to provide privacy + authenticity
- Overhead of resulting protocol small enough that protocols can run on many existing devices

Questions?

http://arxiv.org/abs/1604.06959