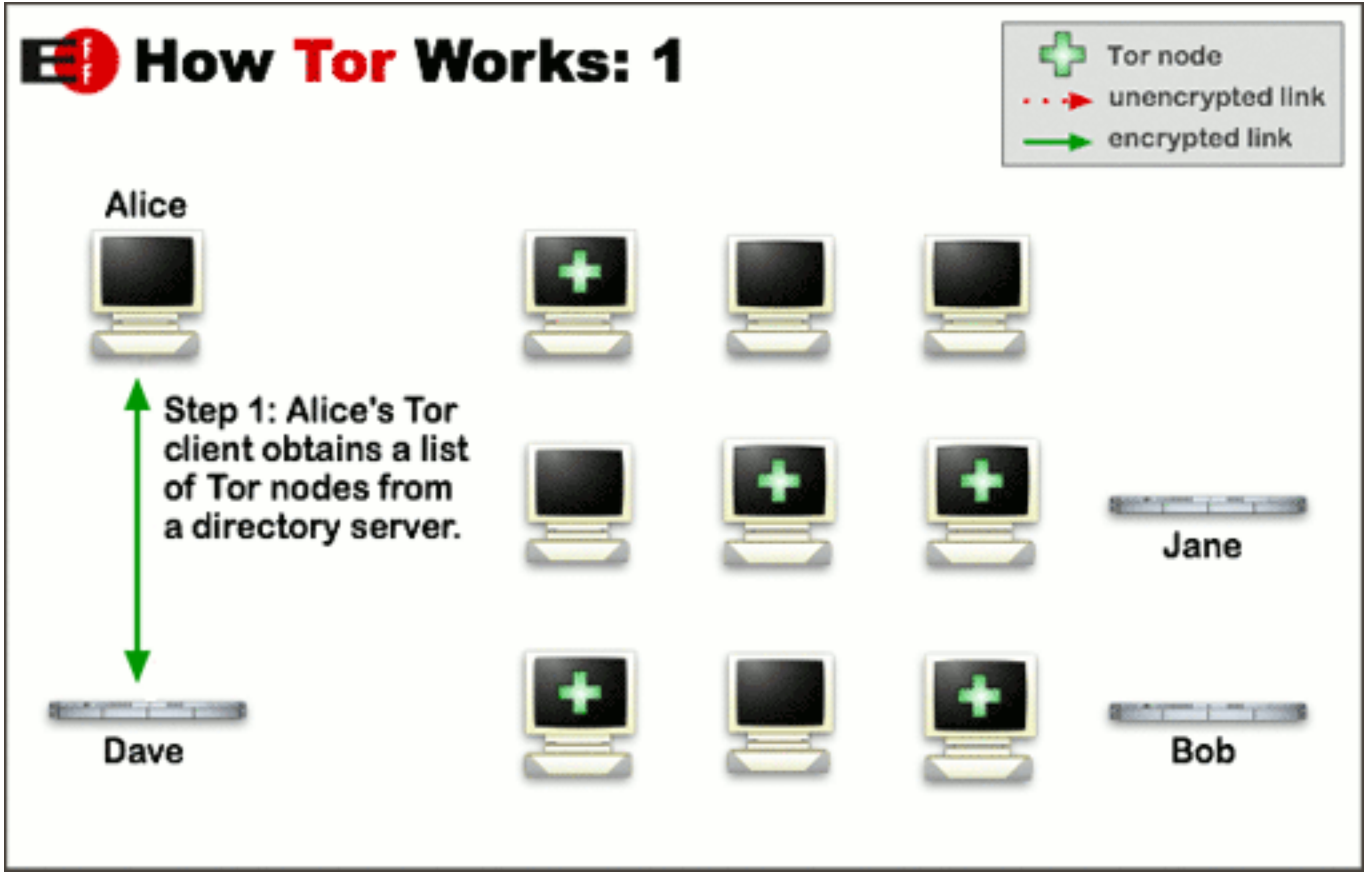
Non-Hidden Hidden Services Considered Harmful

Filippo Valsorda
George Tankersley

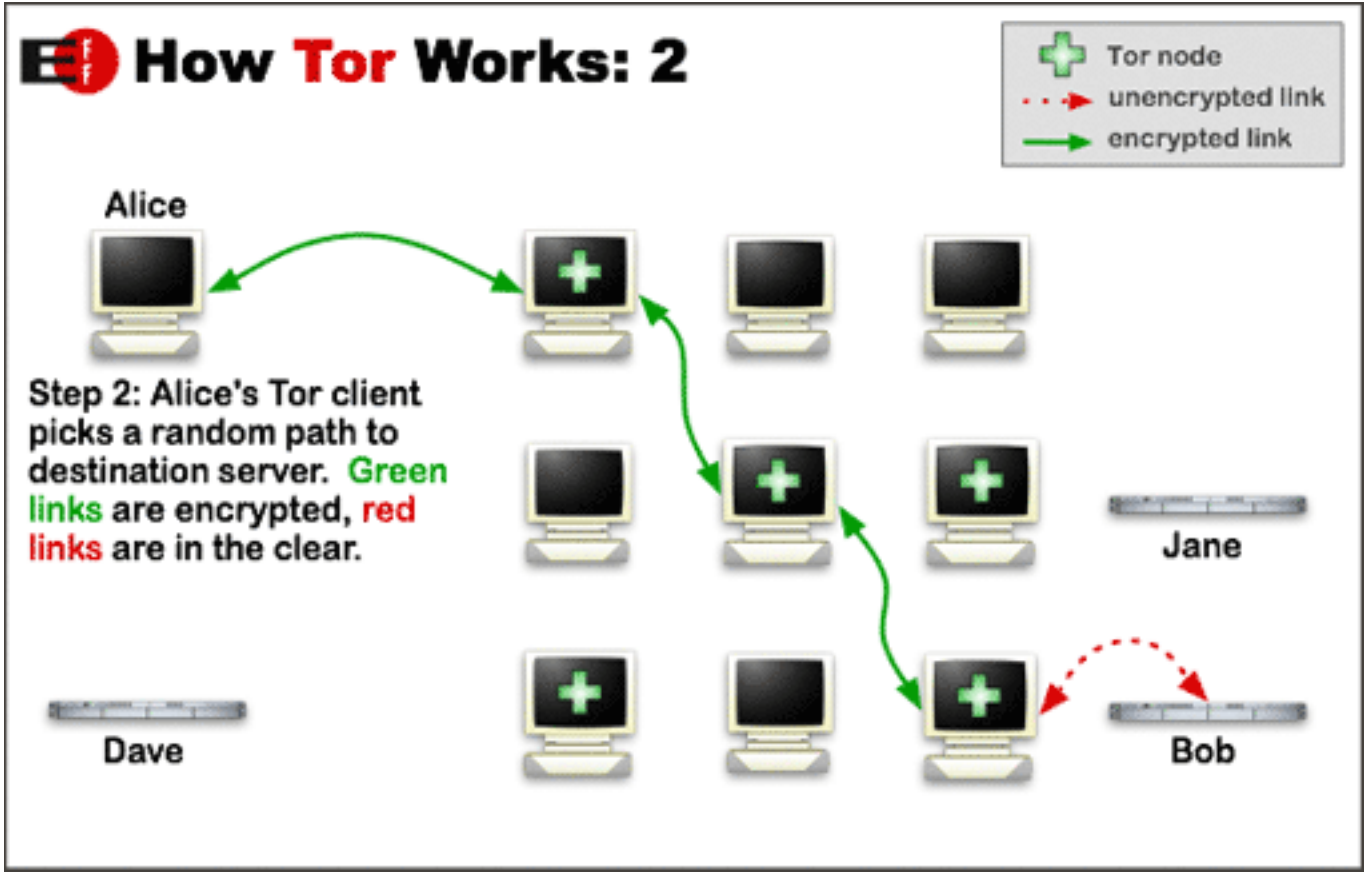
What is Tor?

- **T**he **O**nion **R**outer
- Provides client anonymity
- Works by routing your connection though other machines

Building a circuit


How Tor Works: 1
Tor node
unencrypted link
encrypted link
Alice
Step 1: Alice's Tor client obtains a list of Tor nodes from a directory server.
Jane
Dave
Bob

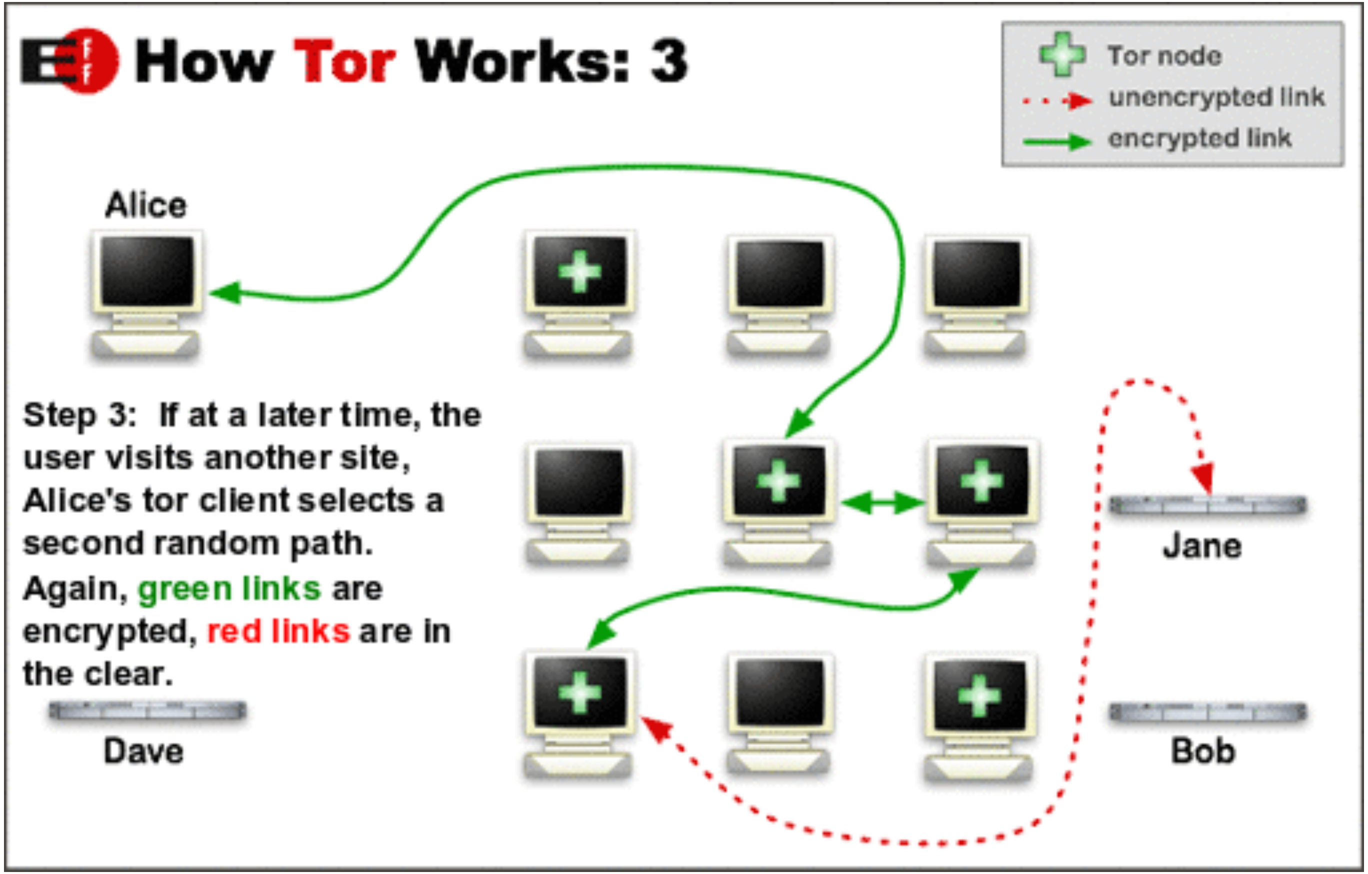
Building a circuit



How Tor Works: 2

Step 2: Alice's Tor client picks a random path to destination server. Green links are encrypted, red links are in the clear.

Building a circuit



How Tor Works: 3

Step 3: If at a later time, the user visits another site, Alice's tor client selects a second random path. Again, green links are encrypted, red links are in the clear.

Hidden Services

- Provide *bidirectional* anonymity

- Supports generic TCP services

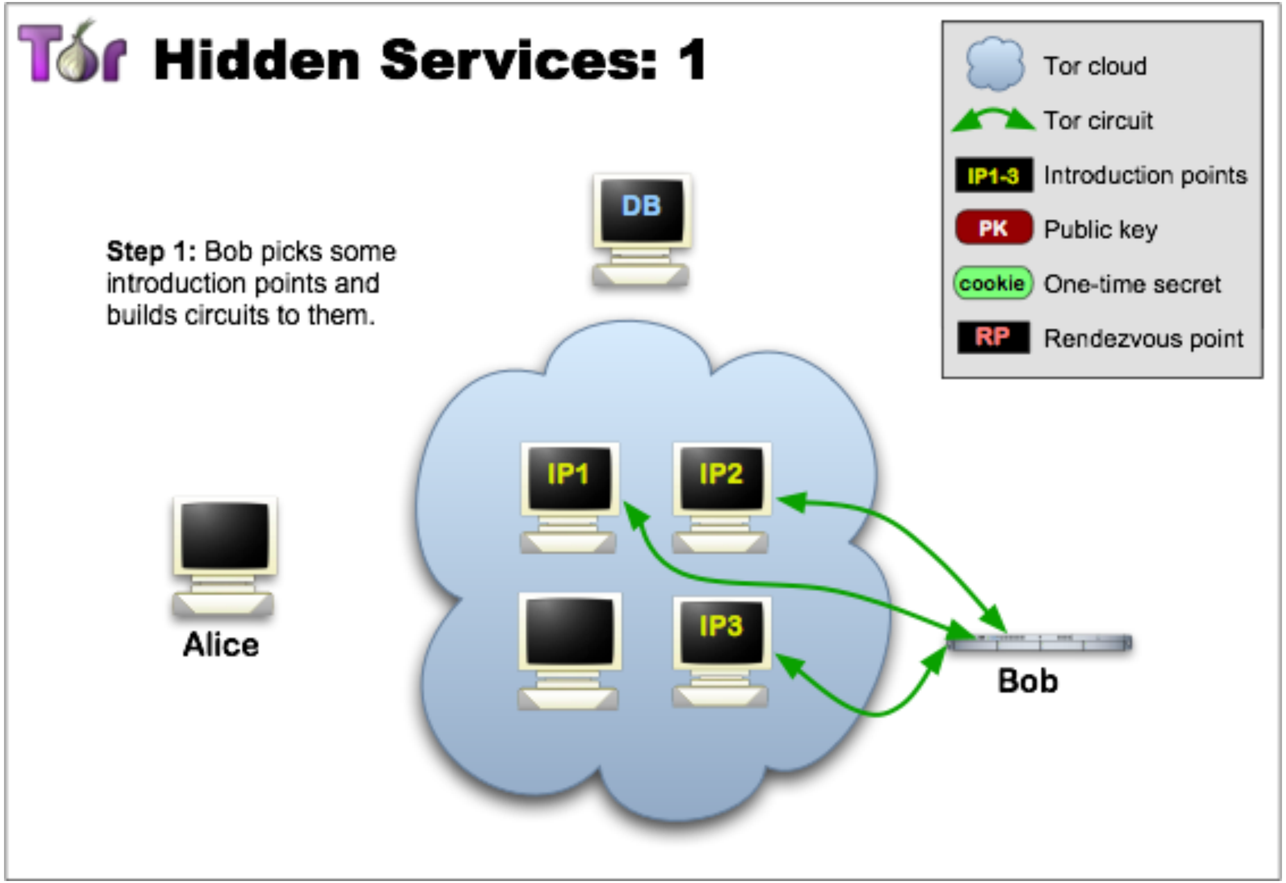
- Famous for drug markets
  - Silk Road
  - Silk Road 2

Hidden Services

But they’re actually used for good

- Whistleblowing (SecureDrop)
- Private chat (Ricochet, XMPP-over-HS)
- Anonymous publishing (of course!)

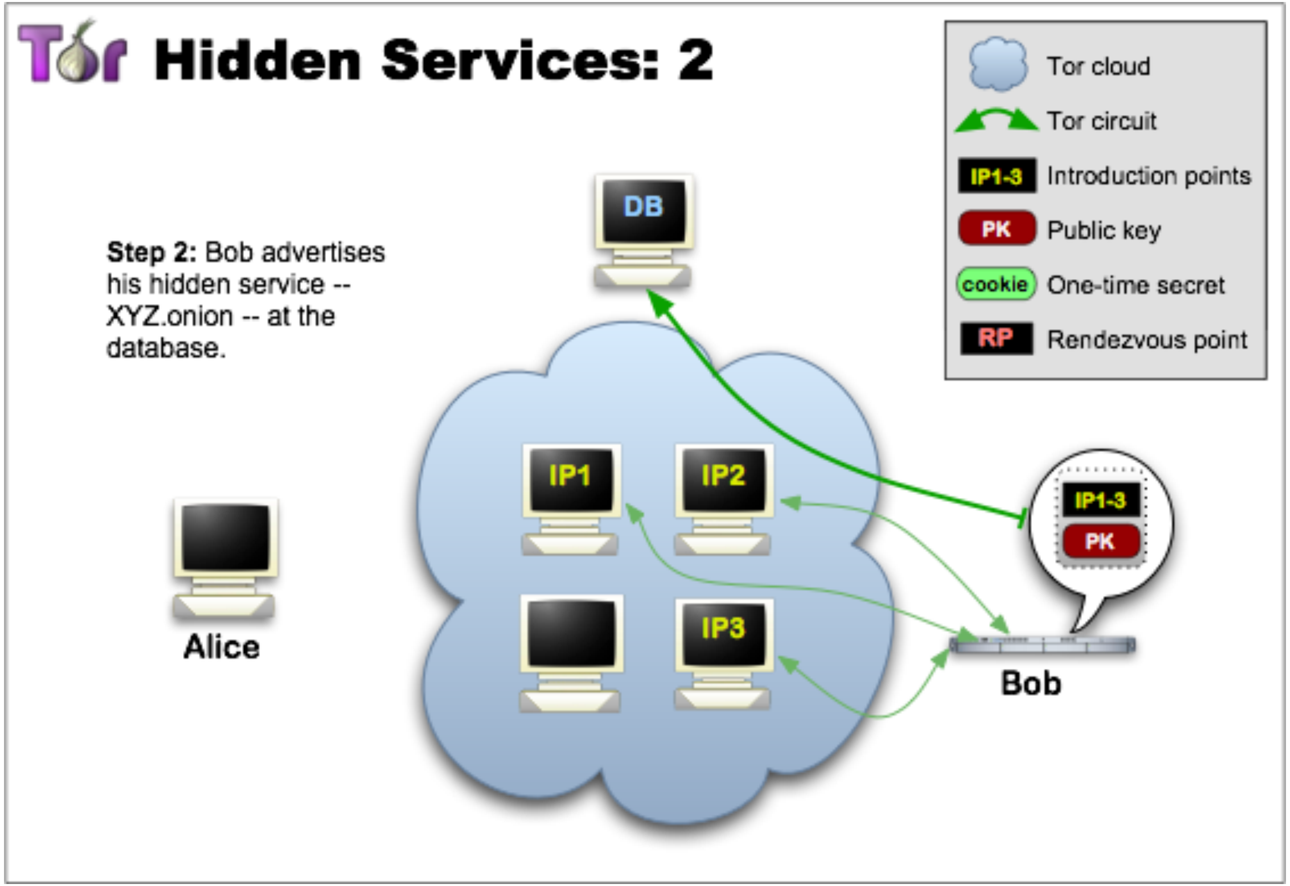
Hidden Services



Tor Hidden Services: 1

Step 1: Bob picks some introduction points and builds circuits to them.

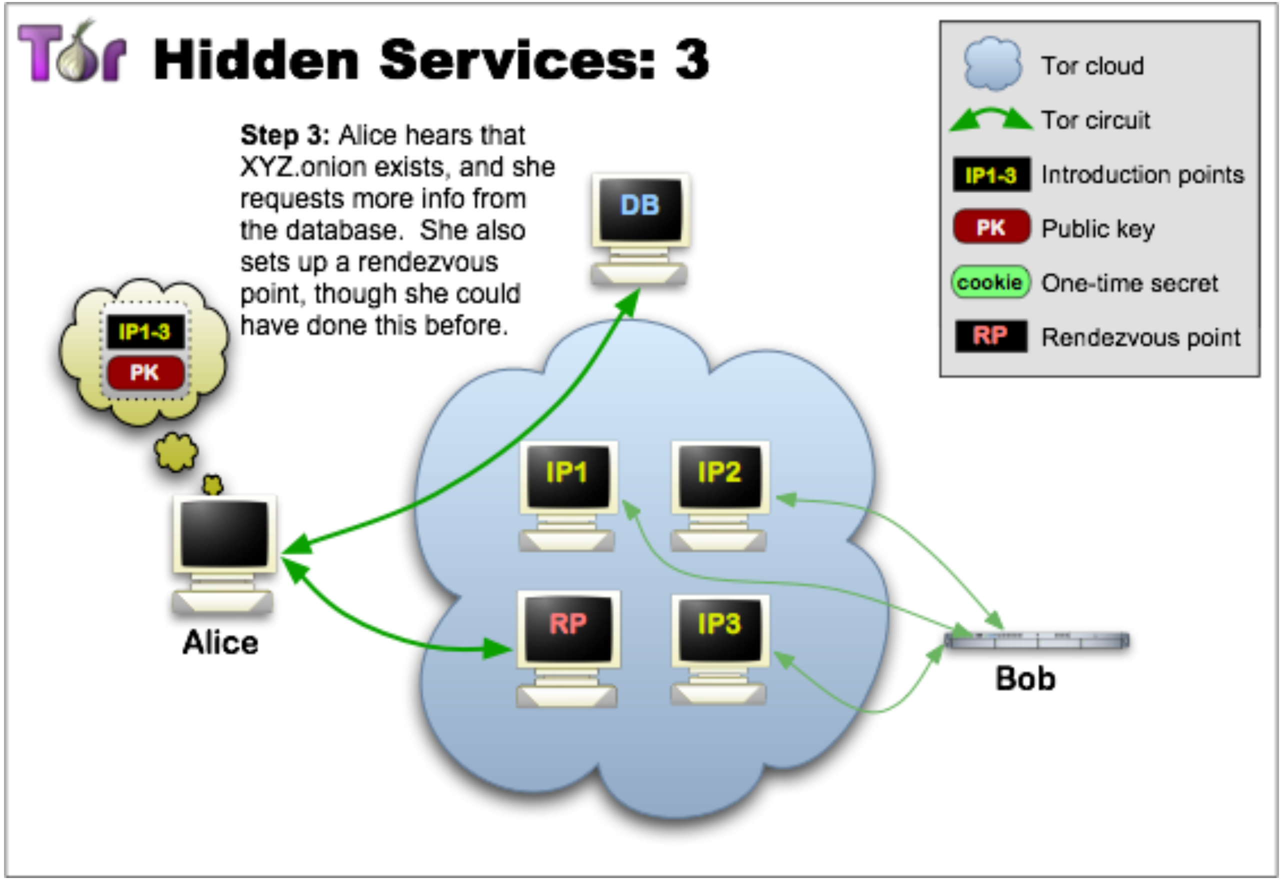
Hidden Services



Hidden Services: 2

Step 2: Bob advertises his hidden service -- XYZ.onion -- at the database.

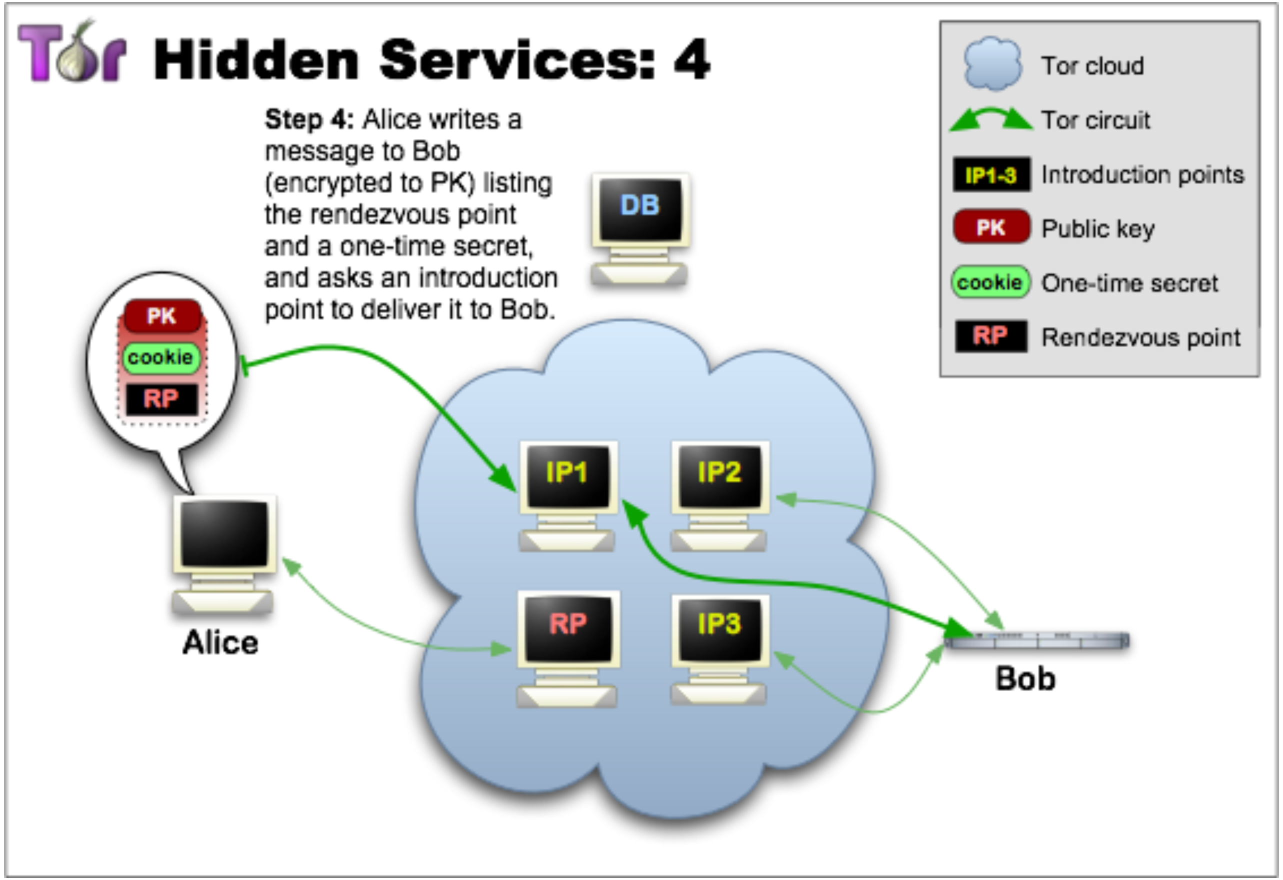
Hidden Services



Tor Hidden Services: 3

Step 3: Alice hears that XYZ.onion exists, and she requests more info from the database. She also sets up a rendezvous point, though she could have done this before.

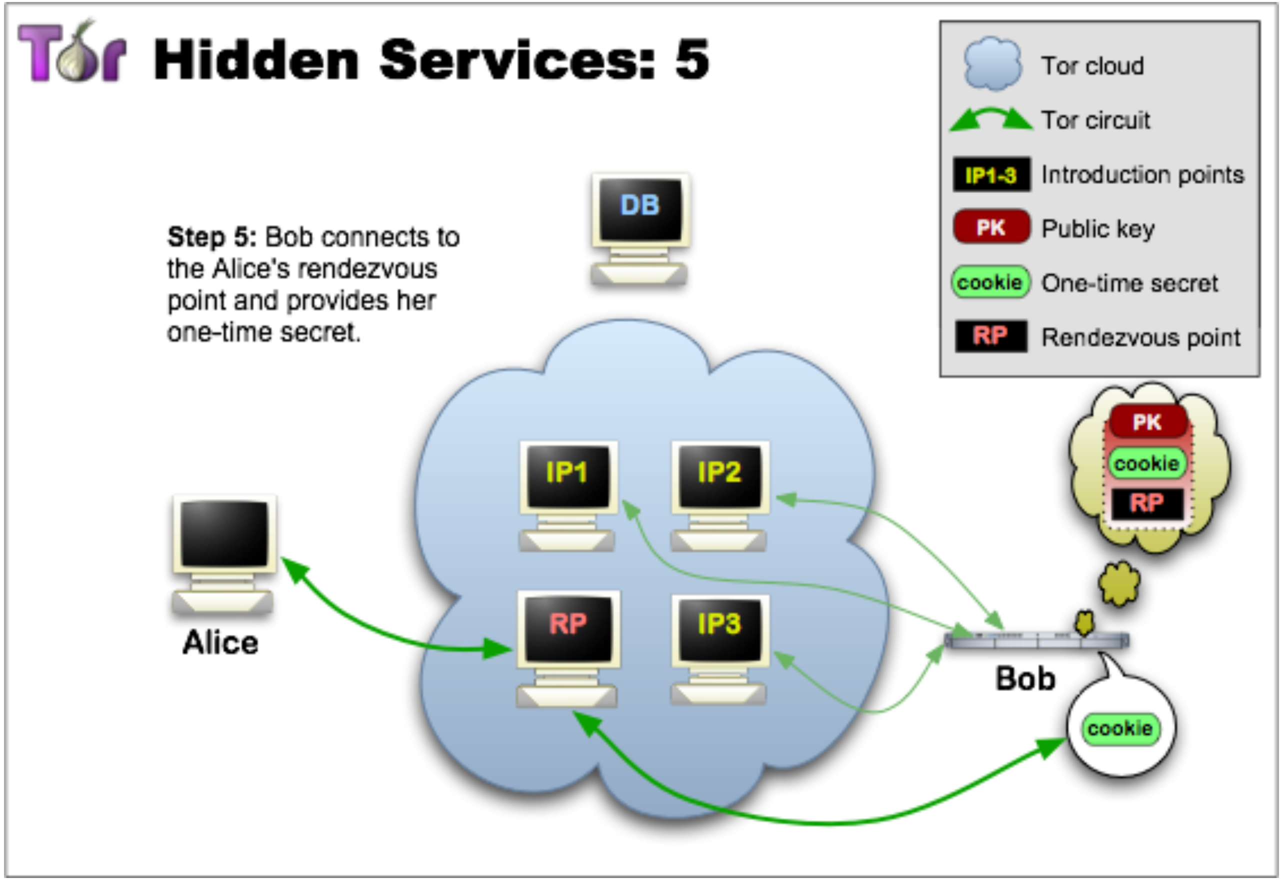
Hidden Services



Tor Hidden Services: 4

Step 4: Alice writes a message to Bob (encrypted to PK) listing the rendezvous point and a one-time secret, and asks an introduction point to deliver it to Bob.

Hidden Services

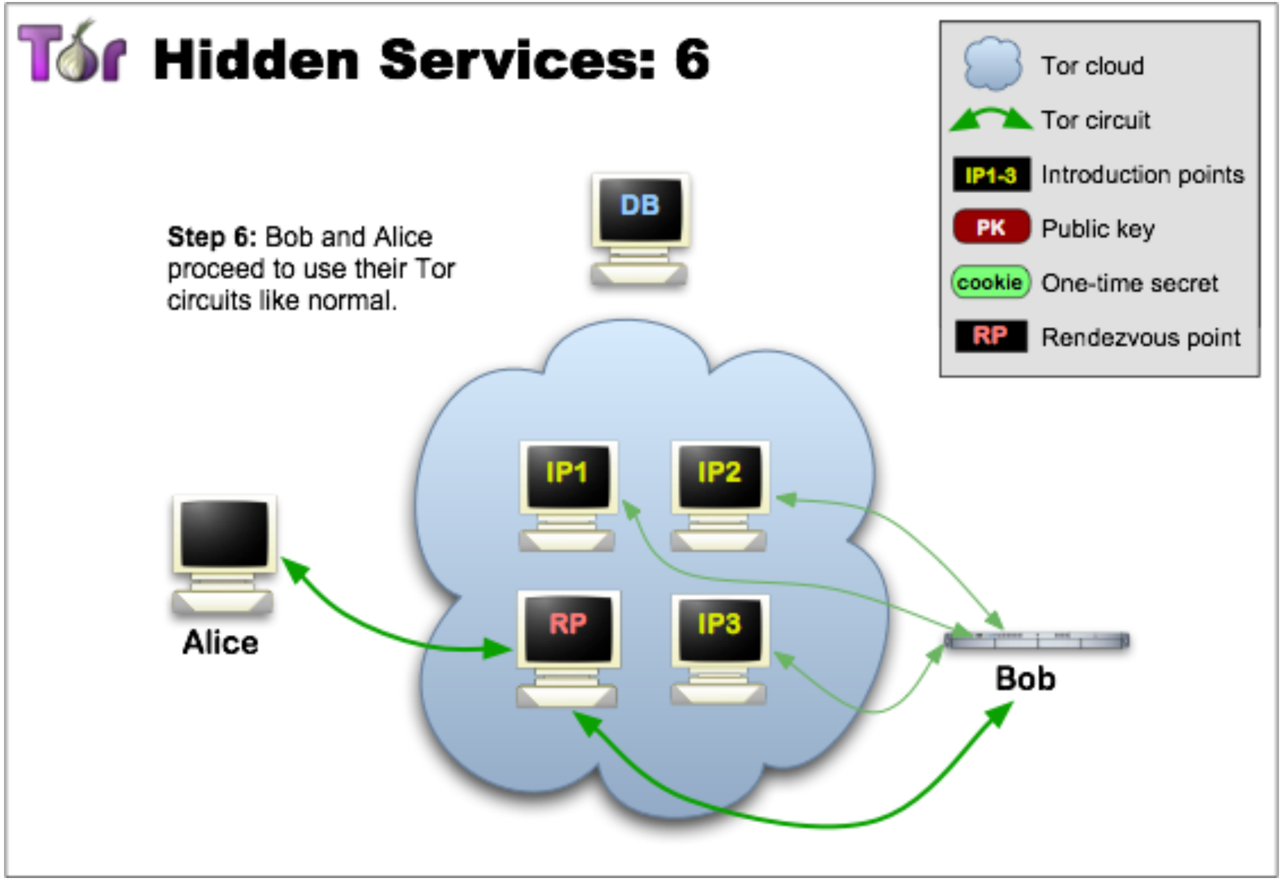


Hidden Services: 5

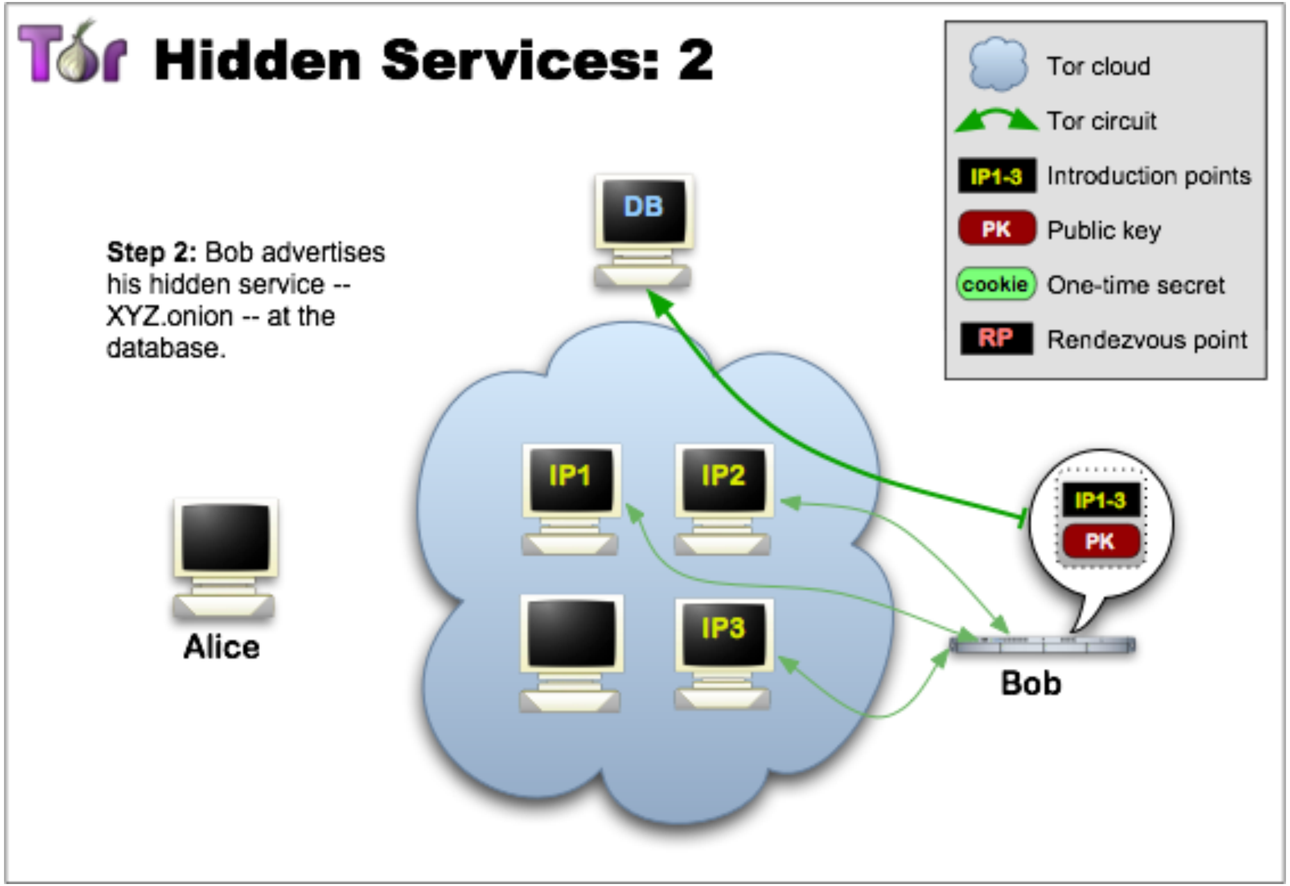
Step 5: Bob connects to the Alice's rendezvous point and provides her one-time secret.

	Tor cloud
	Tor circuit
IP1-3	Introduction points
PK	Public key
cookie	One-time secret
RP	Rendezvous point

Hidden Services



Hidden Services



Tor Hidden Services: 2

Step 2: Bob advertises his hidden service -- XYZ.onion -- at the database.

Hidden Services

The “database” is a DHT made up of stable relays

- directory authorities grant *HSDir* flag
- not related to *Stable* flag

How do we choose where to publish?

HSDir selection

Choose two sets of 3 relays with *HSDir* flag

Think “consistent hashing”

- relays arranged in a ring sorted by identity

Based on a predictable formula (#8244)

HSDir selection

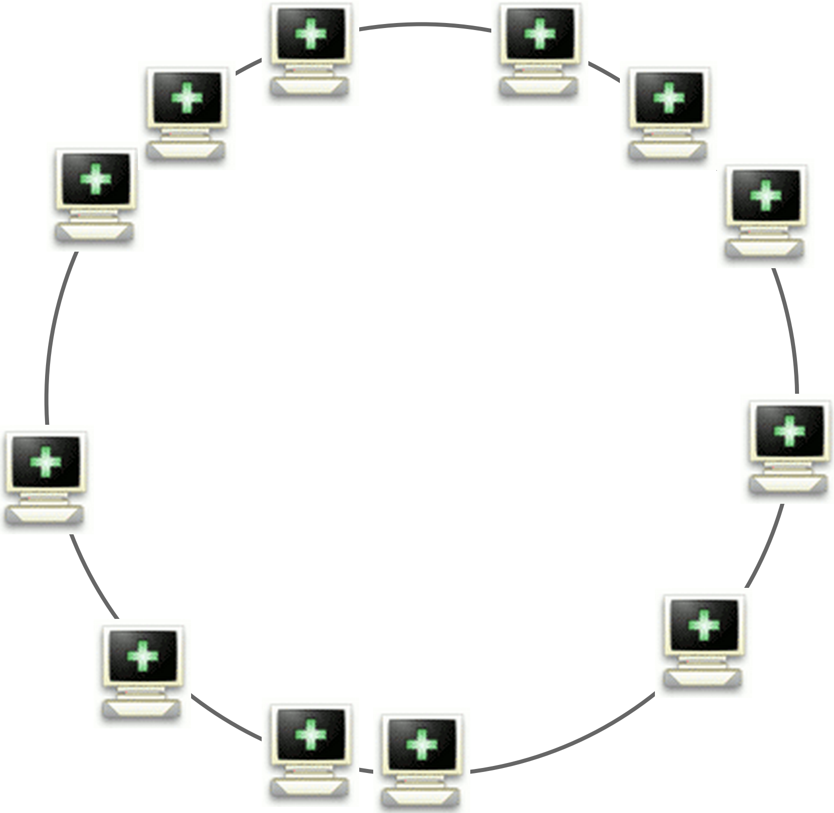
hs-descriptor-id =
    SHA1(id || SHA1(time-period || replica))

id: first 80 bits of SHA1(public key)

time-period: days since epoch (+offset)

replica: which set of HSDirs

HSDir selection



HSDir selection

facebookcorewwwi.onion

descriptor-id =

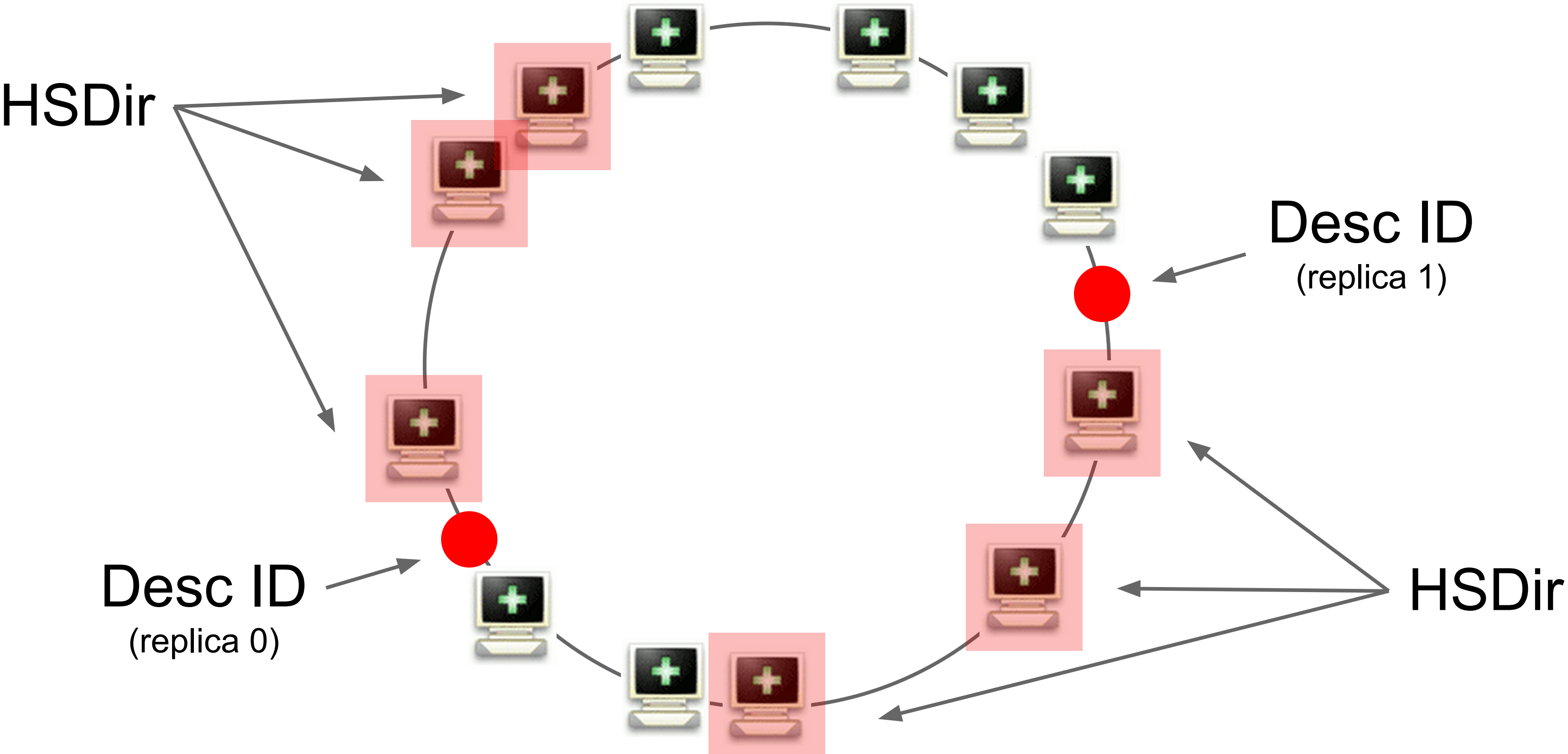
SHA1(facebookcorewwwi || SHA1(16583 || 0))

SHA1(facebookcorewwwi || SHA1(16583 || 1))

replica 0: ys5pml4c6txpw5hnq5v4zn2htytfejf2

replica 1: fq7r4ki5uwcxdxibdl7b7ndvf2mvw2k2

HSDir selection



Why did he just explain all this?

Point of the talk!

Hidden service users face a greater risk of targeted deanonymization than normal Tor users.

Vulnerability of Tor

Low-latency implies correlation attacks

Correlation attacks

in Tor, “both ends” means we’re usually just worried about entry nodes and exit nodes

- **entry nodes** see when a connection starts
- **exit nodes** see when it terminates

Correlation attacks

worried about entry nodes and exit nodes

- ***entry nodes** see when a connection starts*
- ***exit nodes** see when it terminates*

Tor has protections for entry/exit positions

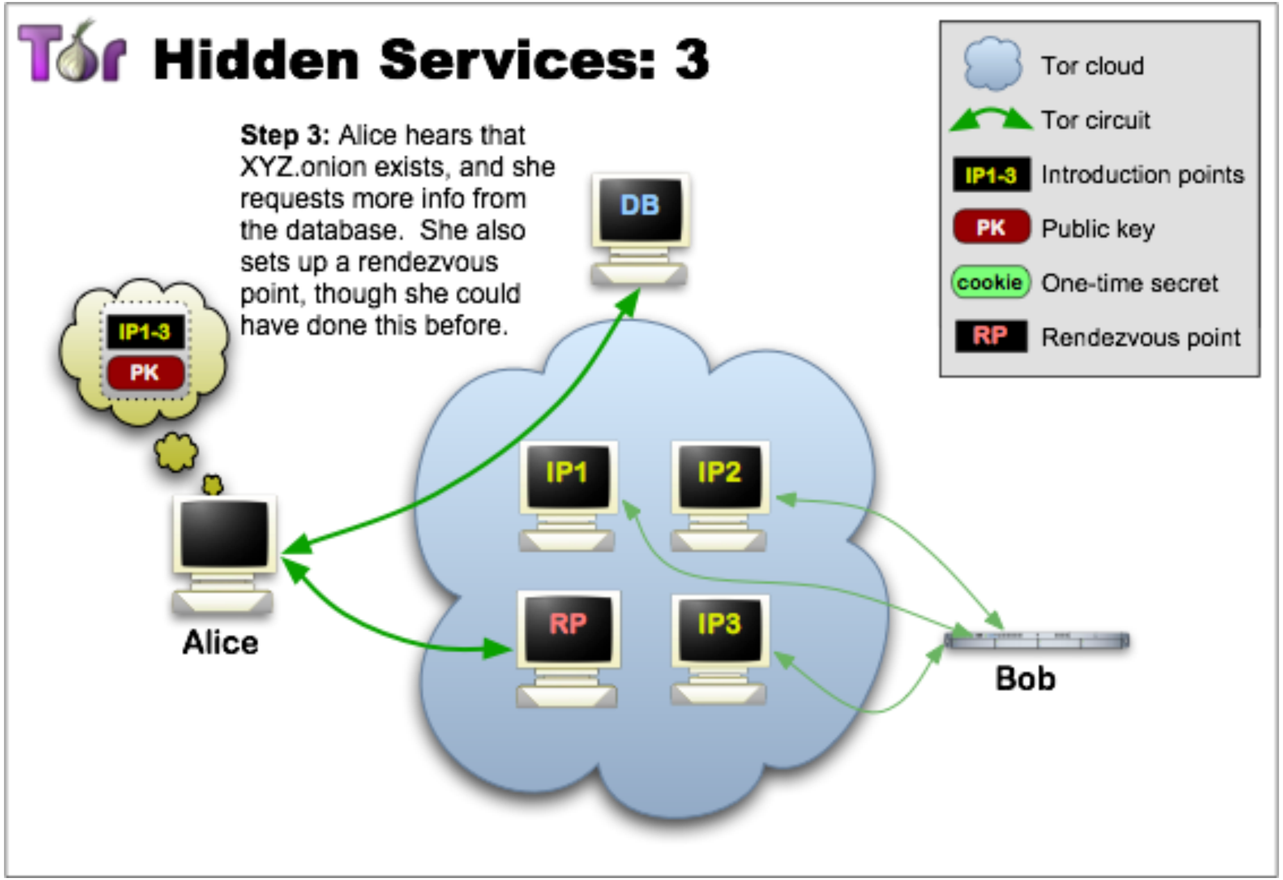
- entry guards, bad relay monitoring, size of network

Correlation attacks

It is hard to become both ends of a circuit.

What else can see when connections happen?

Hidden Services



Hidden Services: 3

Step 3: Alice hears that XYZ.onion exists, and she requests more info from the database. She also sets up a rendezvous point, though she could have done this before.

Hidden Services

An HSDir for a hidden service gets a lookup on ⅙ of requests for information about the hidden service

A lookup indicates a user trying to connect to the hidden service

Correlation attacks

worried about entry nodes and exit nodes

- ***entry nodes** see when a connection starts*
- ***exit nodes** see when it terminates*

For a hidden service, the HSDir can see when a connection happens

Correlation attacks

*worried about entry nodes and **HSDir***

- ***entry nodes** see when a connection starts*
- ***HSDir** see when it terminates*

For a hidden service, the HSDir can see when a connection happens

Correlation attacks

If your target uses a hidden service, don’t need exit relay to see when the connection happens.

Instead, be an HSDir.

Hidden Services

It is very easy to become HSDir

- You just need 4 days uptime
- It should be harder than it is (#8243)

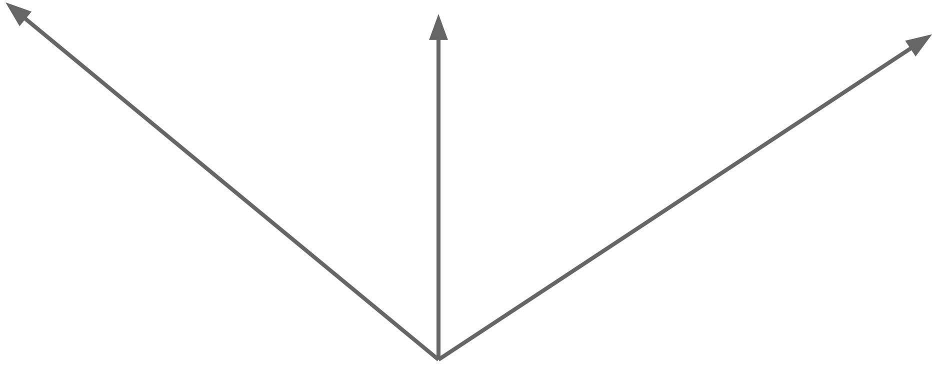
In fact, very easy to become *specific* HSDir

Positioning attack

SHA1(id || SHA1(time-period || replica))

Positioning attack

SHA1(**id** || SHA1(**time-period** || **replica**))



PREDICTABLE

Positioning attack

Predictable and fast? Bruteforce it!

1) Calculate descriptor IDs for the service
2) Generate random 1024-bit RSA key
3) Check if hash precedes the first real descriptor ID in the DHT
4) If not, goto 2

Correlation attacks

If your target uses a hidden service, don’t need exit relay to see when the connection happens.

Instead, be **their** HSDir.

Correlation attacks

If your target uses a hidden service, don’t need exit relay to see when the connection happens.

Instead, be **every** HSDir.

Positioning attack

facebookcorewwwi.onion

descriptor-id =

SHA1(facebookcorewwwi || SHA1(16583 || 0))

SHA1(facebookcorewwwi || SHA1(16583 || 1))

replica 0: ys5pml4c6txpw5hnq5v4zn2htytfejf2

replica 1: fq7r4ki5uwcxdxibdl7b7ndvf2mvw2k2

HSDirs should have been

Fingerprint	Nickname
C4F205C1024779B663584BBDFEB3F9C3C7689750	aoiharu
C4F2B201A09F8D72EFE2648C0B998249E9B95D15	ovce
C514A3E6D98385E47BA6D67C632383A549C1C115	CherryBomb
2C40E3C8B254A3F20064E7914F8A39FF3DE1CCC0	jantor
2C4488ECDE14563D25DA3D1A8B172C4E547F4CD8	RebelOnion1
2C4E15CD40EE3D2D6F062F04ADFE9B85C8C3C52B	Unzane

HSDirs actually were

Fingerprint	Nickname
C4BF08CE48880453DC0E9186AF2B4922BB275380	unduplicablerelay
C4C8DF4DDFCFAB2936C6F07E91D7D6AF07A6E147	EquaTOR
C4E108F2C98F4B60BA9EE560DD928296632D4389	Unnamed
2C3FC687783A4F1E9AA098EB8762F8FF7331C2DD	mushroomMUSHROOM
2C40B4194C26857A7A26E6B9E8D0C63E40600A1C	penguinxtor
2C40E3C8B254A3F20064E7914F8A39FF3DE1CCC0	jantor

HSDirs actually were

Fingerprint	Nickname
C4BF08CE48880453DC0E9186AF2B4922BB275380	unduplicablerelay
C4C8DF4DDFCFAB2936C6F07E91D7D6AF07A6E147	EquaTOR
C4E108F2C98F4B60BA9EE560DD928296632D4389	Unnamed
2C3FC687783A4F1E9AA098EB8762F8FF7331C2DD	mushroomMUSHROOM
2C40B4194C26857A7A26E6B9E8D0C63E40600A1C	penguinxtor
2C40E3C8B254A3F20064E7914F8A39FF3DE1CCC0	jantor

HSDirs actually were

Fingerprint	Nickname
C4BF08CE48880453DC0E9186AF2B4922BB275380	unduplicablerelay
C4C8DF4DDFCFAB2936C6F07E91D7D6AF07A6E147	EquaTOR
C4E108F2C98F4B60BA9EE560DD928296632D4389	**Unnamed**
2C3FC687783A4F1E9AA098EB8762F8FF7331C2DD	mushroomMUSHROOM
2C40B4194C26857A7A26E6B9E8D0C63E40600A1C	penguinxtor
2C40E3C8B254A3F20064E7914F8A39FF3DE1CCC0	jantor

Vulnerability of Tor

worried about entry nodes and HSDir

- ***entry nodes** see when a connection starts*
- *HSDir see when it terminates*

Vulnerability of Tor

worried about entry nodes and HSDir

- ***many people** see when a connection starts*
- *HSDir see when it terminates*

Vulnerability of Tor

worried about entry nodes and HSDir

- ***many people** see when a connection starts*
- *HSDir see when it terminates*

“entry” does not just mean your entry node

- ISP, malicious access point, pen register…

Summarizing all of that

1) HSDirs can serve the same purpose against a hidden service as a malicious exit relay would in a basic correlation attack

2) The “entry side” of a Tor connection can be monitored by means other than compromising guards

Summarizing all of that

It’s actually ***worse***, because it’s way easier to be the user’s HSDir.

Hidden service users face a greater risk of targeted deanonymization than normal Tor users.

Corollary

If you run a hidden service that does not need location hiding, you are unnecessarily exposing your users to this risk.

It would probably be better to let them use Tor on your TLS-enabled clearnet site.

There is hope

Proposal #224 is “Next-Generation Hidden Services”

Go read it and help out if you can!

https://tinyurl.com/hidserv

In the meantime: defense!

HS operators can do this.

You can trust an HSDir you run yourself.

With some safety margin:

6 nodes * 5 days = 30

with 2 nodes per IP, 15 machines (rolling buffer)

In the meantime: defense!

HS operators can do this.

You can trust an HSDir you run yourself.

Free detection: you will notice if someone competes with you for the HSDir positions.

In the meantime: detection!

Hidden service operators should watch HSDirs

What makes a suspicious HSDir?

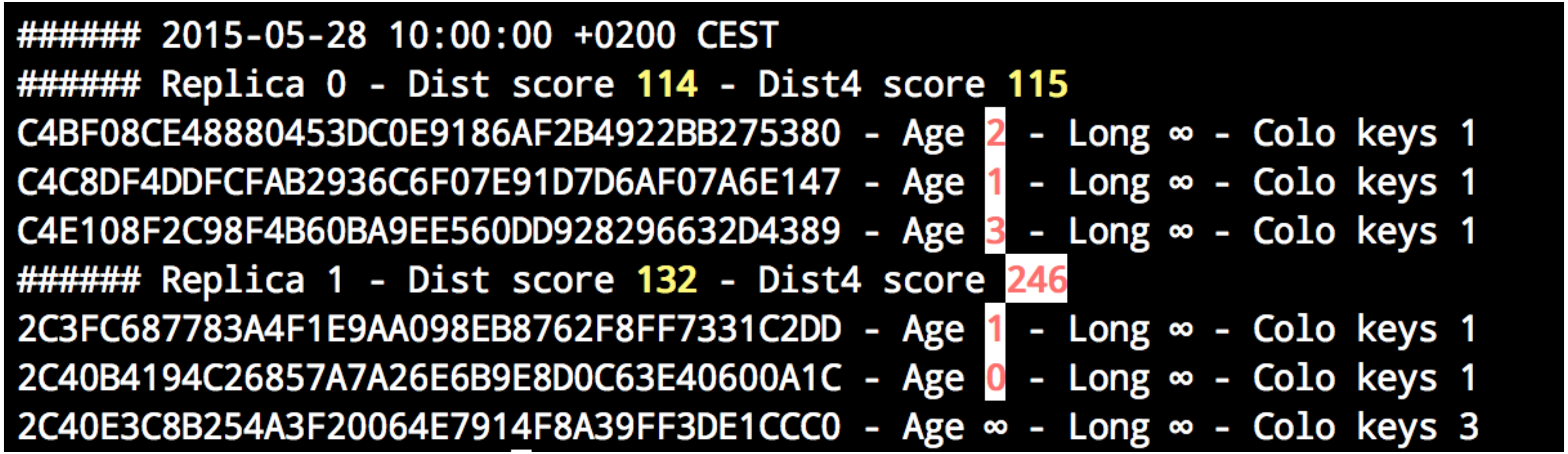
Suspicious HSDir metrics

- Dense fingerprints
- Low age
- Low longevity after the HSDir event
- Many keys seen on the same (or related) IP

- And maybe other stuff! AS? Clustering?

Suspicious HSDir metrics

We made tools for this: https://hsdir.org



```
####### 2015-05-28 10:00:00 +0200 CEST
####### Replica 0 - Dist score 114 - Dist4 score 115
C4BF08CE48880453DC0E9186AF2B4922BB275380 - Age 2 - Long ∞ - Colo keys 1
C4C8DF4DDFCFAB2936C6F07E91D7D6AF07A6E147 - Age 1 - Long ∞ - Colo keys 1
C4E108F2C98F4B60BA9EE560DD928296632D4389 - Age 3 - Long ∞ - Colo keys 1
####### Replica 1 - Dist score 132 - Dist4 score 246
2C3FC687783A4F1E9AA098EB8762F8FF7331C2DD - Age 1 - Long ∞ - Colo keys 1
2C40B4194C26857A7A26E6B9E8D0C63E40600A1C - Age 0 - Long ∞ - Colo keys 1
2C40E3C8B254A3F20064E7914F8A39FF3DE1CCC0 - Age ∞ - Long ∞ - Colo keys 3
```

Questions?

https://hsdir.org

Filippo Valsorda (@FiloSottile)
filippo@cloudflare.com

George Tankersley (@_gtank)
george.tankersley@coreos.com