Today in Cryptography (5830)

Crypto backdoors

Cryptographic backdoors

- Long debate over whether average citizens should have access to strong crypto
  - “Crypto wars” of 1990s: export restrictions that treat crypto software as munitions

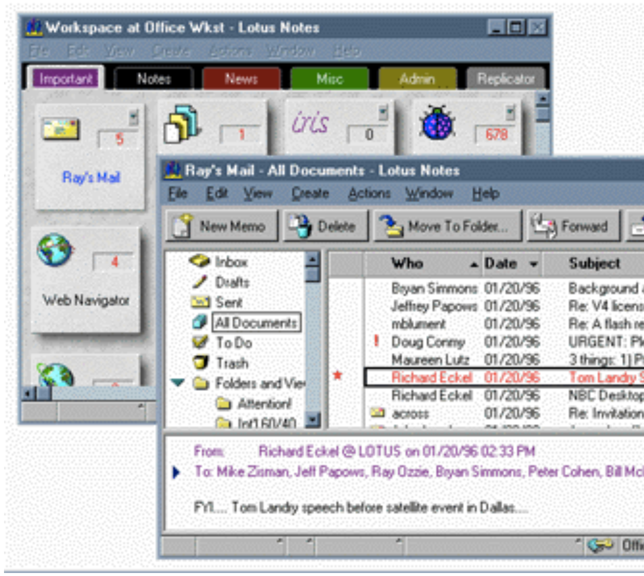
- Overt and surreptitious backdoors seen as backup plan by governments

Overt backdoors

- Clipper chip
  - NSA hardware for encrypting telecommunications
  - Each chip had secret key, this was given to (escrowed with) NSA at manufacture time
- Significant backlash
- “The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption” by Abelson et al.

Overt backdoors

- Export controls required only 40-bit keys for international software
- Lotus Notes “Differential Workfactor Cryptography”
  - 64 bit symmetric key K
  - C1 = RSA-Enc(pk_{NSA} , top24(K))
  - C2 = Enc(K, data)



Surreptitious backdoors

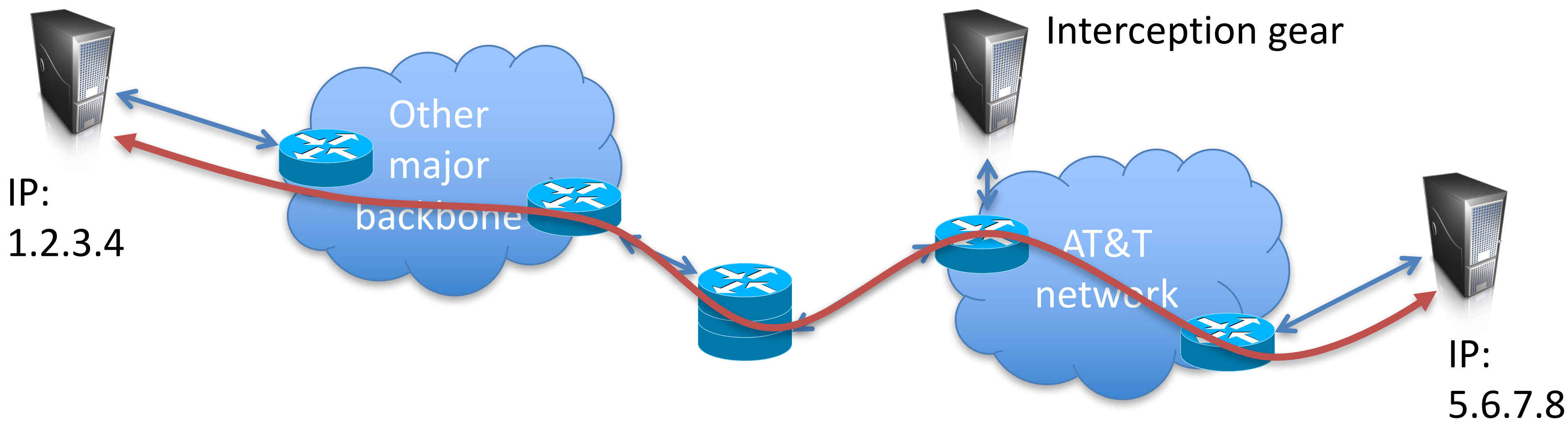
- Secretly weaken / sabotage cryptographic systems
- Usually done to dovetail with interception capabilities

AT&T Wiretap case

- Mark Klein discloses potential wiretapping activities by NSA at San Francisco AT&T office
- Fiber optic splitter on major trunk line for Internet communications
  - Electronic voice and data communications copied to “secret room”
  - Narus STA 6400 device

Preventing intercept

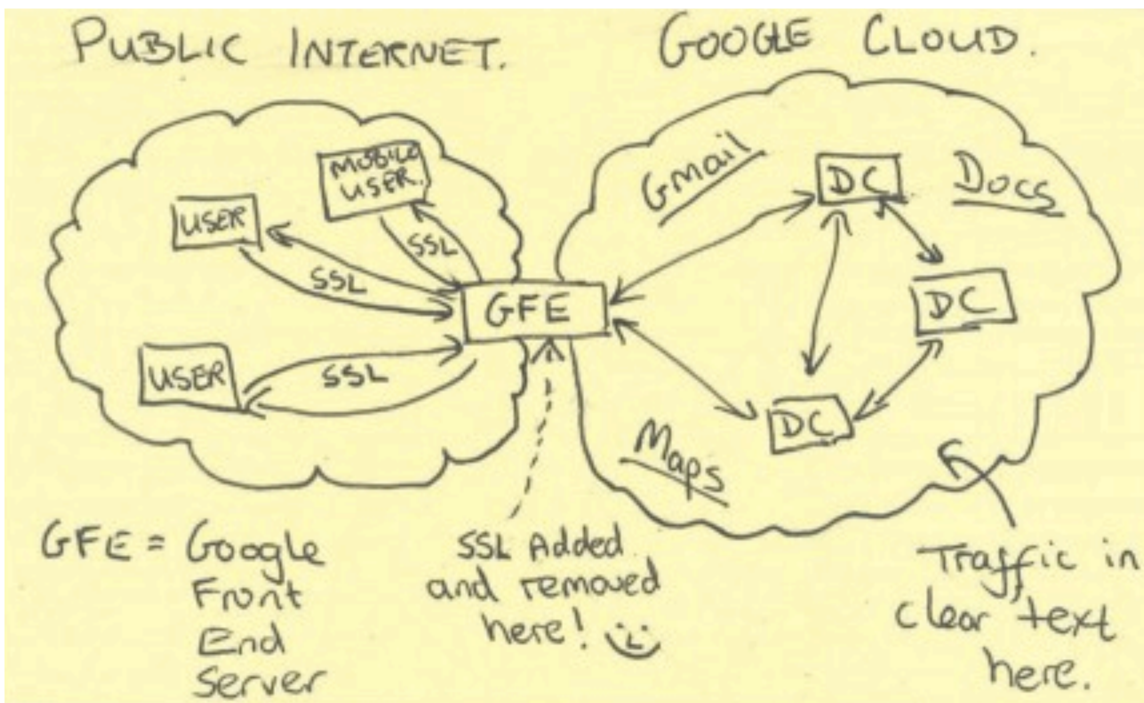
- End-to-end encryption (TLS, SSH)



- What does this protect? What does it leak?
- What can go wrong?

End-run around HTTPS

- HTTPS terminated at edge of Google networks
- Internal data center-to-data center communications on privately leased lines
  - No encryption up until summer 2013



Sabotage of crypto

- Surveillance would benefit from sabotage of cryptographic protocols / implementations
- Revelations indicate NSA sought to accomplish this
  - Dual EC PRNG case probably most well known

Desiderata for good sabotage:

- Allow decryption, ideally in real time
- Decryption should be private
  - Only saboteur should be able to exploit
- Undetectability

- Others?

See [Schneier et al. 2015] for taxonomy and easy-to-read summary

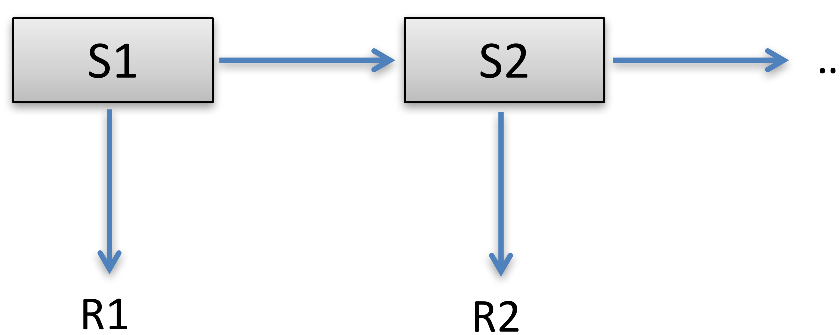
Sabotaging PRNGs

- Say we can sabotage client's random numbers to make them predictable
- Where do random numbers come from?
  - Use system service like /dev/urandom to generate initial seed S1
  - Use S with a pseudorandom number generator (PRNG)

(S2, R1) <- PRG(S1)

(S3, R2) <- PRG(S2)

⋮



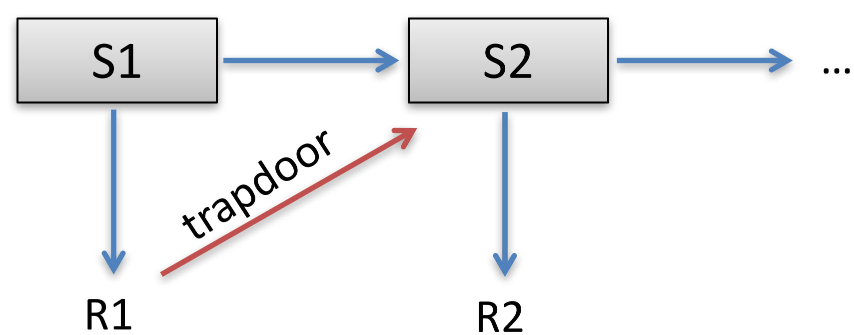
Sabotaging the PRNG

- Arrange that given R1, attacker with a trapdoor can compute S2
- This allows predicting all subsequent values

(S2, R1) <- PRG(S1)

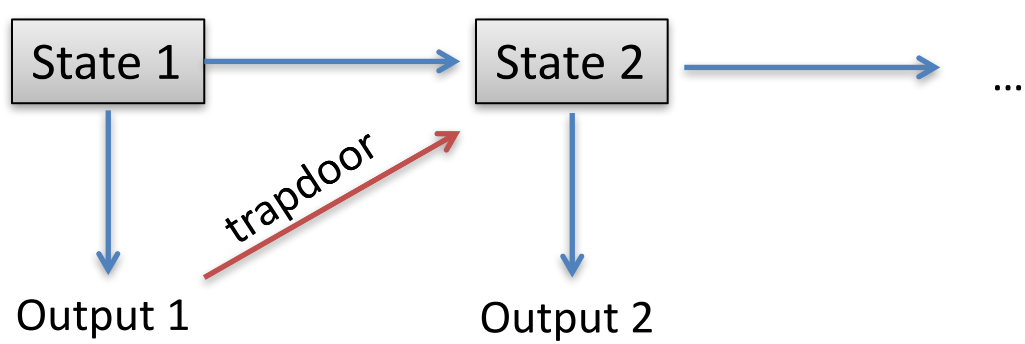
(S3, R2) <- PRG(S2)

⋮



Sabotaging PRNGs

- NIST's Dual EC pseudorandom number generator (PRNG) apparently backdoored
  - Mandated public parameters are public key
  - There exists a secret key, the trapdoor
- One output of PRNG + trapdoor reveals next state of PRNG, and prediction of future outputs



Bank customer

TLS handshake for RSA transport

Bank

Pick random Nc

ClientHello, MaxVer, Nc, Ciphers/CompMethods →

Pick random Ns

← ServerHello, Ver, Ns, SessionID, Cipher/CompMethod

← CERT = (pk of bank, signature over it)

Check CERT using CA public verification key

Pick random PMS
C <- E(pk,PMS)

C →

PMS <- D(sk,C)

Say client is using Dual EC for randomness generation

What is vulnerable?

RSA BSAFE library: 2.4 seconds to recover PMS

Windows: 60 minutes

OpenSSL: never (bug in code!)

See http://dualec.org/

A Simple Diffie-Hellman Trapdoor

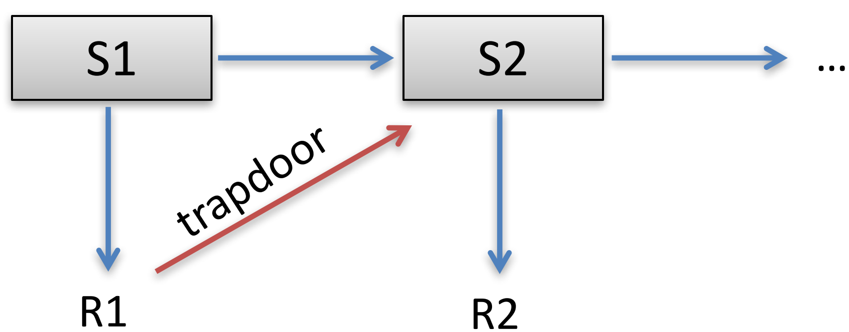
Let G be a cryptographically strong group with generator g

Let P in G be chosen parameter. Choose to be $P = g^p$

Let seed S1 be uniform value in $\mathbf{Z}_{|G|}$

$$PRG(S1) = \left(H(P^{S1}),\ g^{S1} \right) = (S2, R1)$$

Given R1, p, compute $S2 = H(R1^p)$



Can view R1 as public-key encryption of next seed S2

Good PRNG to anyone without trapdoor p

Dual EC is very similar

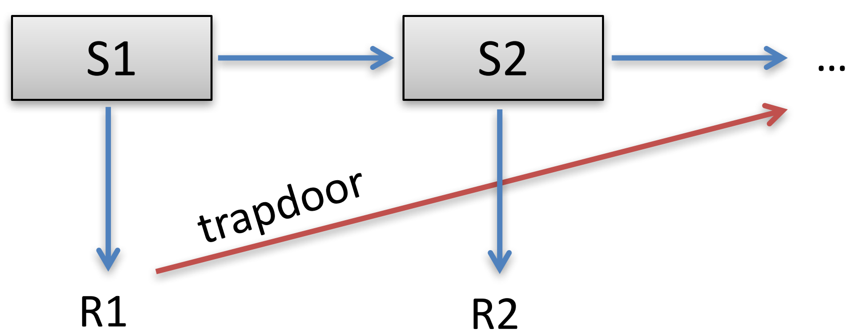
Let G be a cryptographically strong group with generator g

Let P in G be chosen parameter. Choose to be $P = g^p$

Let seed S1 be uniform value in $\mathbf{Z}_{|G|}$

$$PRG(S1) = (P^{S1}, g^{S2}) = (S2, R1)$$
$$PRG(S2) = (P^{S2}, g^{S3}) = (S3, R2)$$

Given R1, p, compute $S3 = R1^p = g^{S2*p} = P^{S2}$



Actually, truncates 16 bits from R1. Can brute-force

How good is this backdoor?

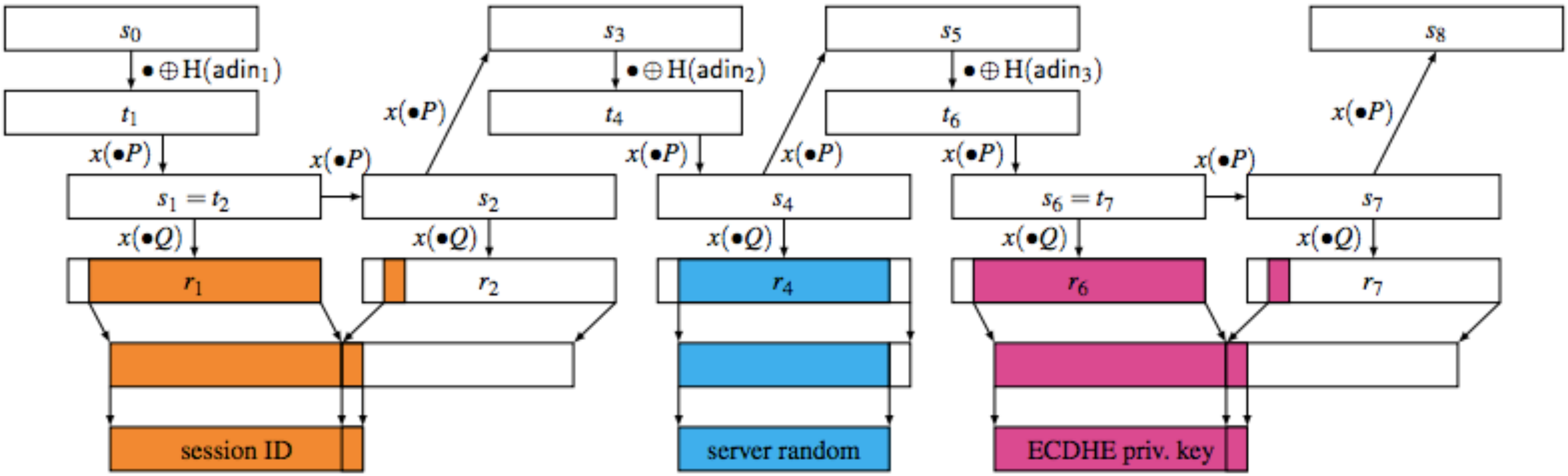
- Undetectability:
  - Shumow, Fergusen discovered backdoor in 2005, while Dual EC went through standardization process
  - Standardized anyway...
- Effectiveness:
  - PRNG may not be used in exploitable ways
  - May not be used in first place (many faster PRNGs out there)
  - More bits of R1 may be truncated
  - May be implemented incorrectly
  - Dual EC supports *additional inputs* that could add entropy, making attacks harder

Checkoway et al. 2014 study

- Investigate implementations of TLS:

  openssl, Windows schannel, RSA BSAFE
- Conclude that some are more vulnerable than others:
  - Openssl bug prevents use of Dual EC (easy to fix)
  - Windows schannel uses additional input (deviates from Dual EC spec in ways that make attack faster)
  - RSA BSAFE very vulnerable

Checkoway et al. 2014 study



From [Checkoway et al. 2014]. Diagram of Dual EC use within openssl (after bug is fixed)

Checkoway et al. 2014 study

Library	Default PRNG	Extended Random	Bytes per Session	Additional Entropy	Time (minutes)
BSAFE C	✓		31–60	—	0.04
BSAFE Java	✓	✓	28	—	63.96
SChannel I			28	—	62.97
SChannel II			30	—	182.64
OpenSSL-fixed I			32	20	0.02
OpenSSL-fixed II			32	35	83.32
OpenSSL-fixed III			32	$35+k$	$2^k \cdot 83.32$

ZNet scan of IPv4: only 720 servers using BSAFE Java

Juniper Dual EC Incident

[Checkoway et al. 2016]

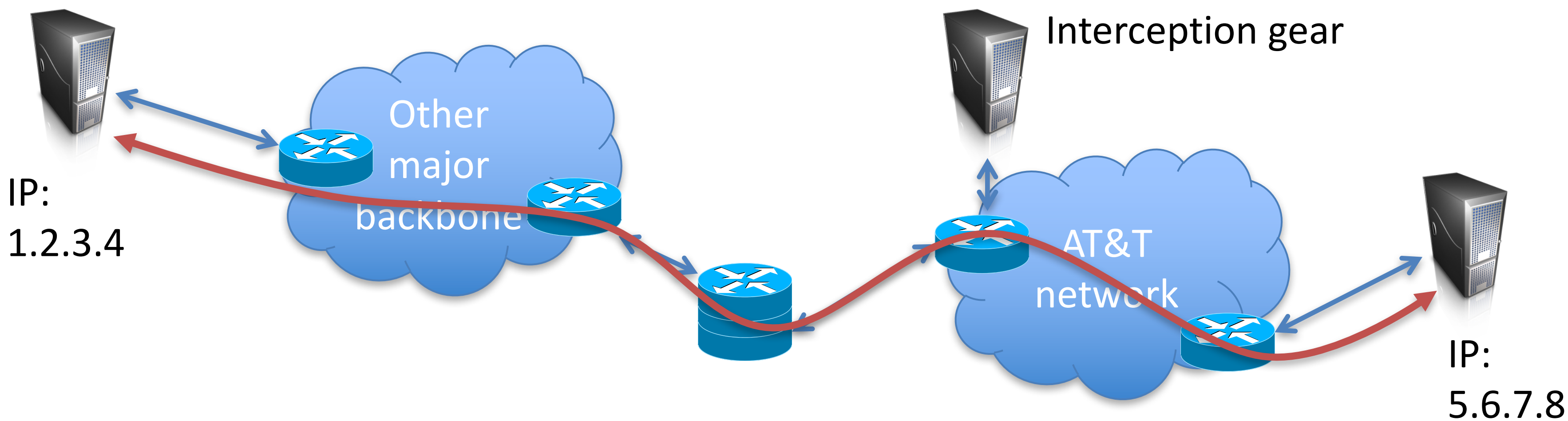
- ScreenOS used in Juniper NetScreen firewall products. Used to perform VPN encryption
- Uses Dual EC, but supposedly wrapped within another PRNG. Shouldn’t be vulnerable, even to someone with trapdoor
- But it was. Worse, someone broke in and modified P to a new value P’.
- Single 2008 patch modified P, introduced bug disabling secondary PRNG

Policy debate ongoing

- "Going dark" debate over last few years
  - Police and others argue encryption is preventing criminals, terrorists from being caught
  - Push for building in backdoors into crypto & other systems
  - Manhattan DA have interesting report about smartphone unlocking
- Cryptographers & security folks argue that mandated backdoors are really bad idea
  - Keys under doormats report

Preventing intercept

- End-to-end encryption (TLS, SSH)



- What does this protect? What does it leak?
- What can go wrong?

Hiding “metadata” such as connectivity is hard

- IP addresses are required to route communication, yet not encrypted by normal end-to-end encryption
  - 1.2.3.4 talked to 5.6.7.8 over HTTPs
- How can we hide connectivity information?

Anonymization systems

- Single-hop proxy services

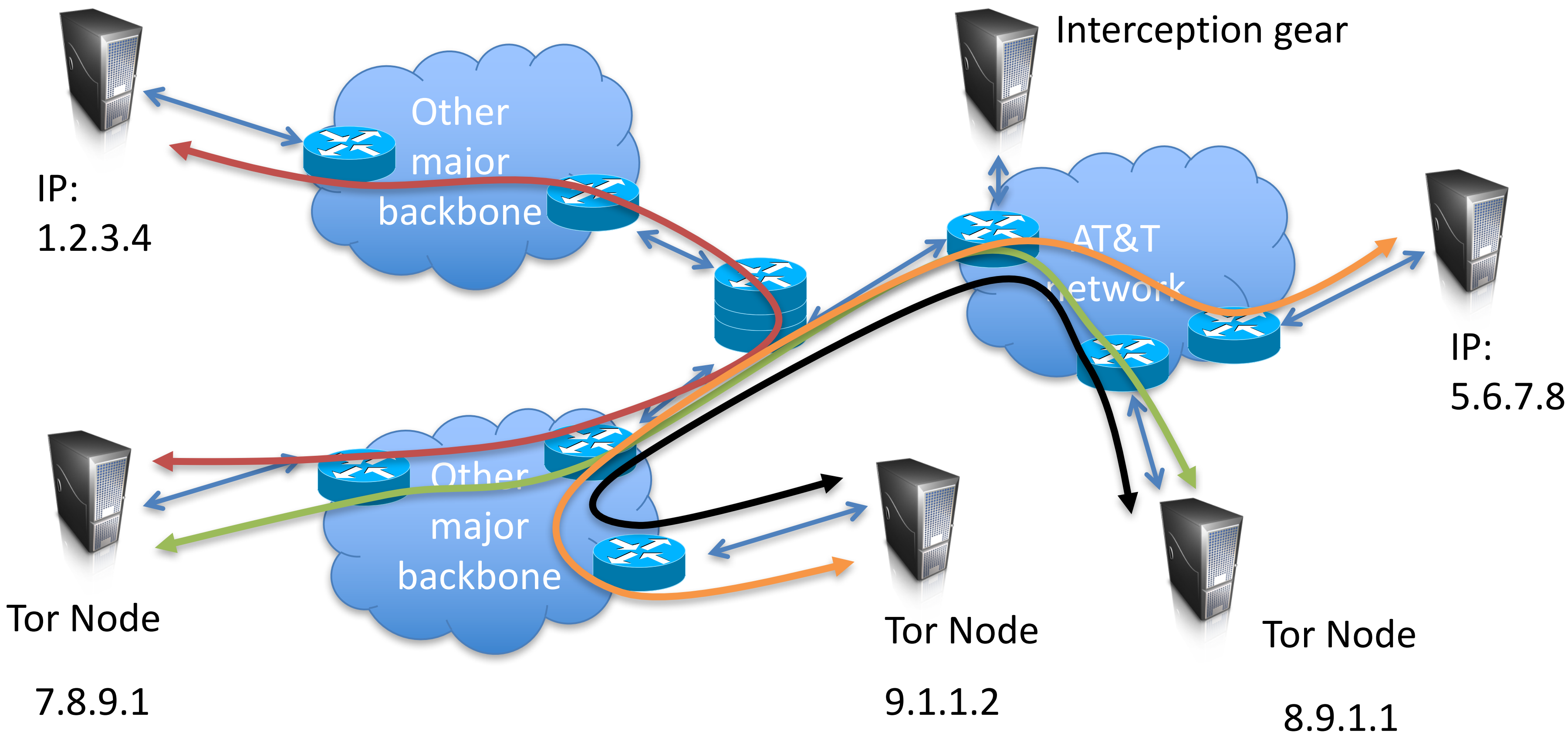
- JonDonym, anonymous remailers (MixMaster, MixMinion), many more…

Thursday, April 26, 2012

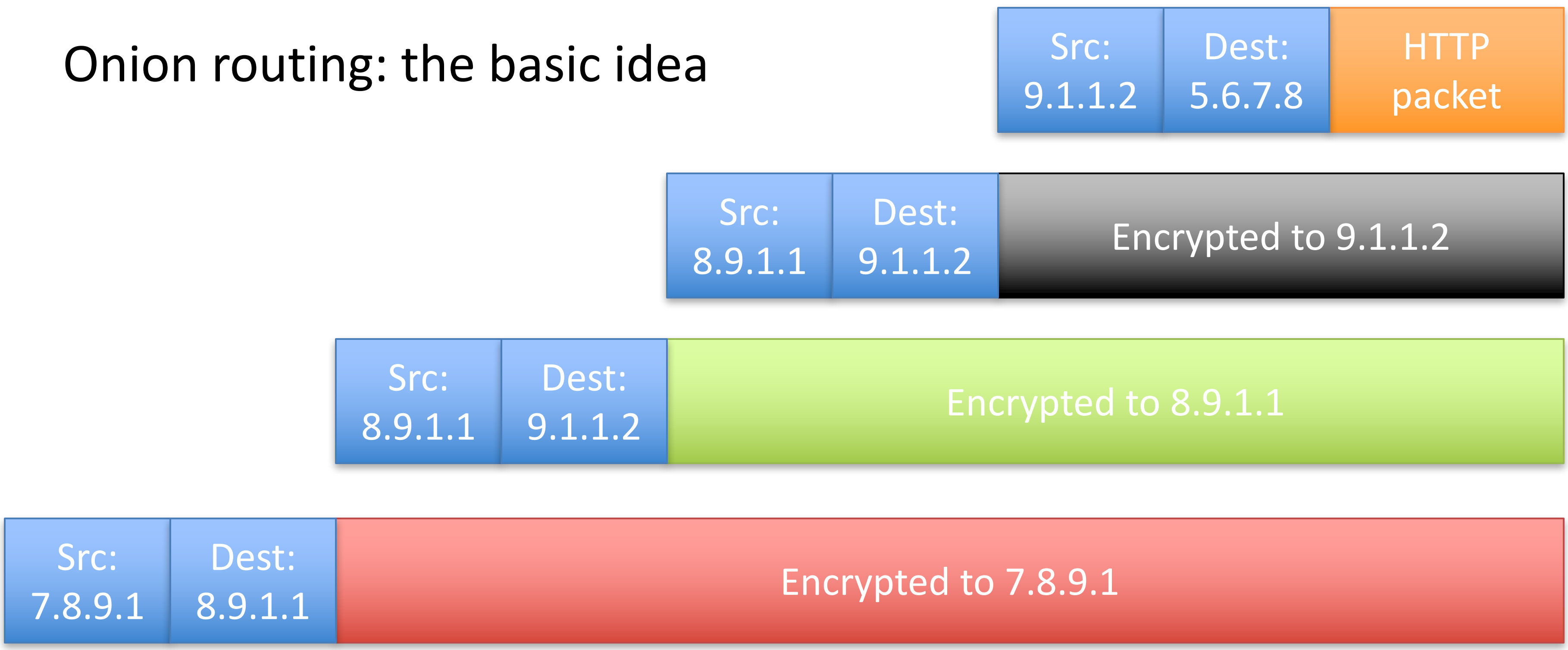
FBI seizes server used to anonymize e-mail

Jeffrey Brown 1 comment

Tor (The Onion Router)

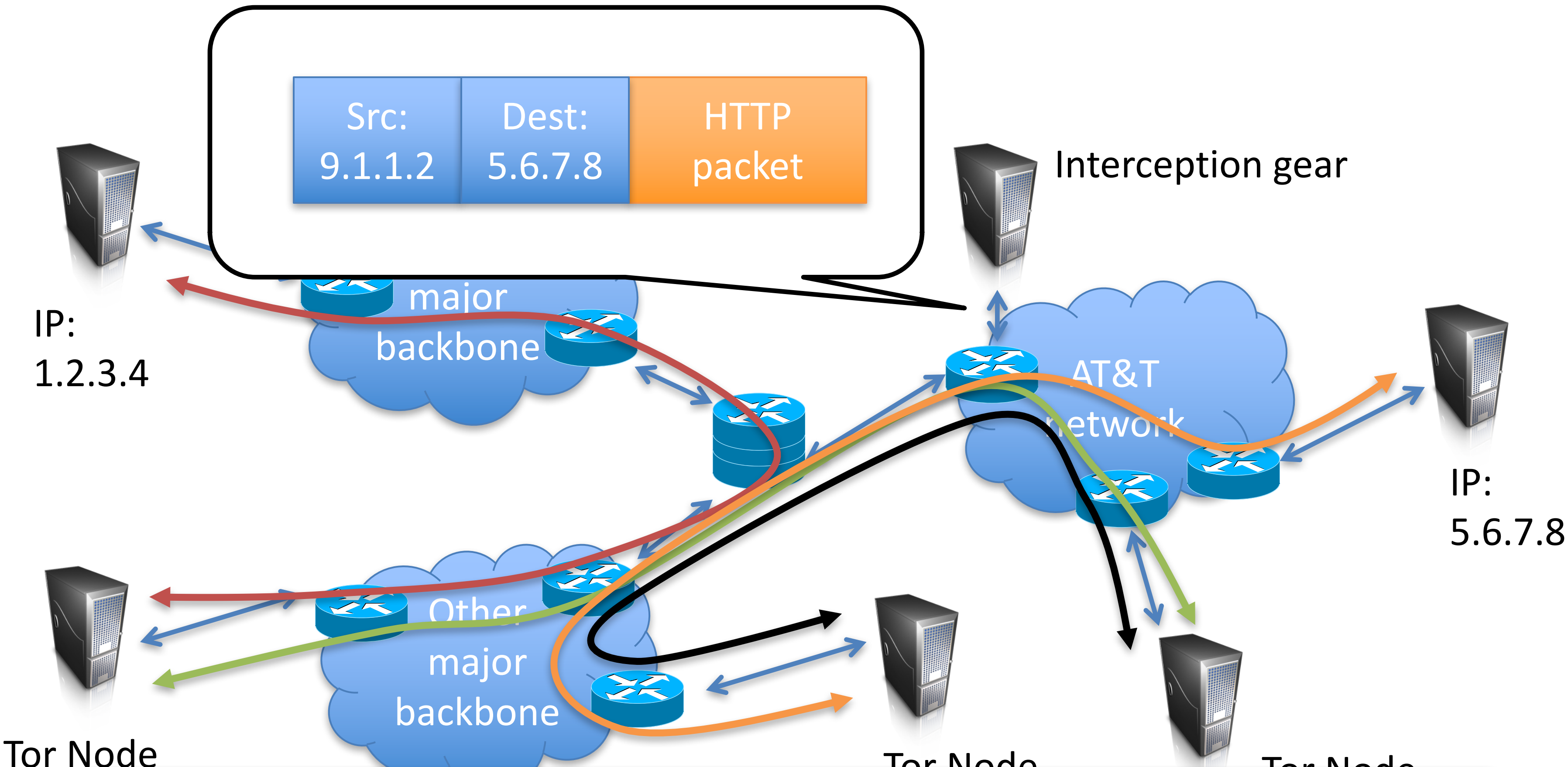


Onion routing: the basic idea



Tor implements more complex version of this basic idea

What does adversary see?



Tor obfuscates who talked to who, need end-to-end encryption (e.g., HTTPS) to protect payload