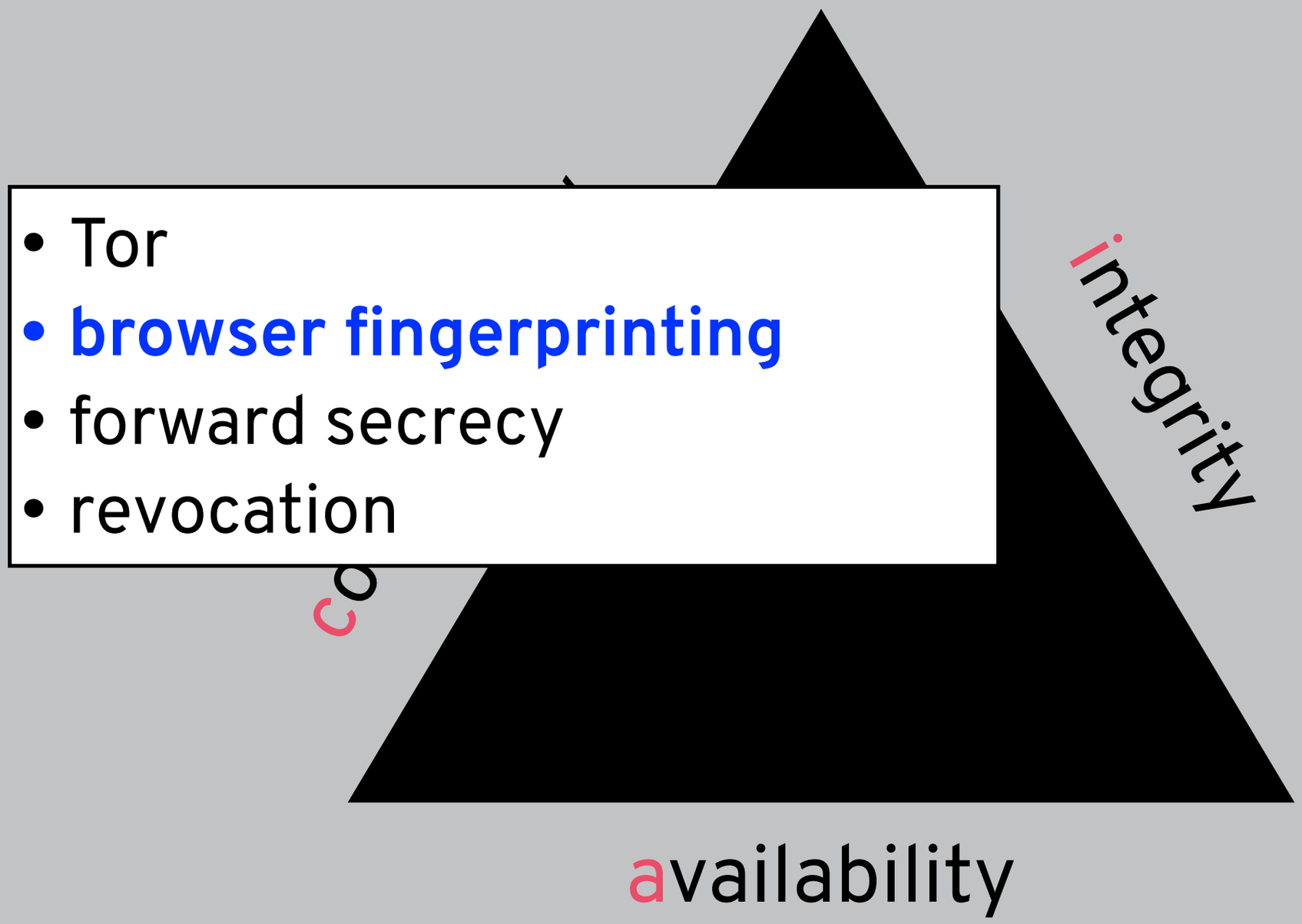
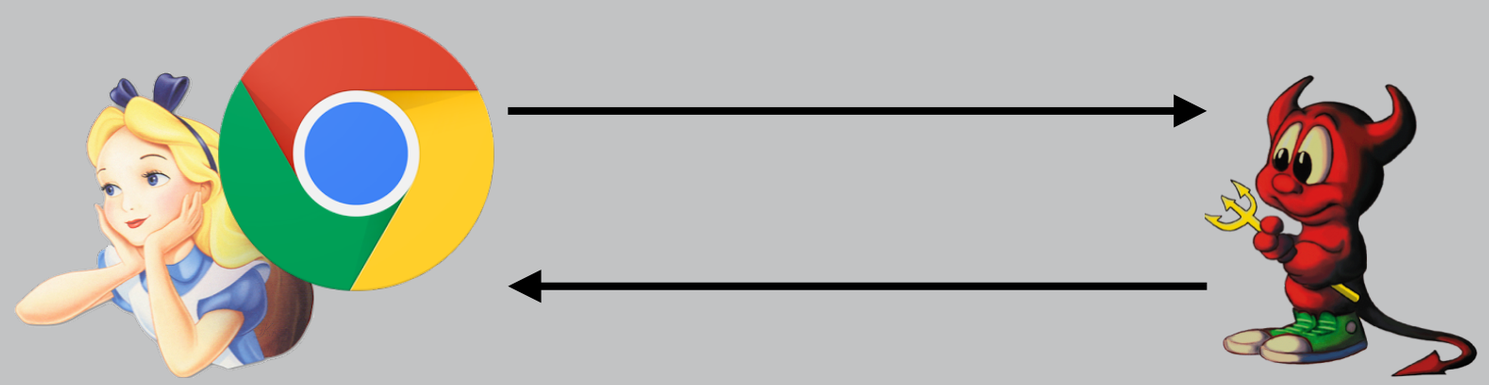
SECURITY (COMP0141): CONFIDENTIALITY ON THE WEB

CONFIDENTIALITY, REVISITED



- Tor
- **browser fingerprinting**
- forward secrecy
- revocation

THREAT MODEL



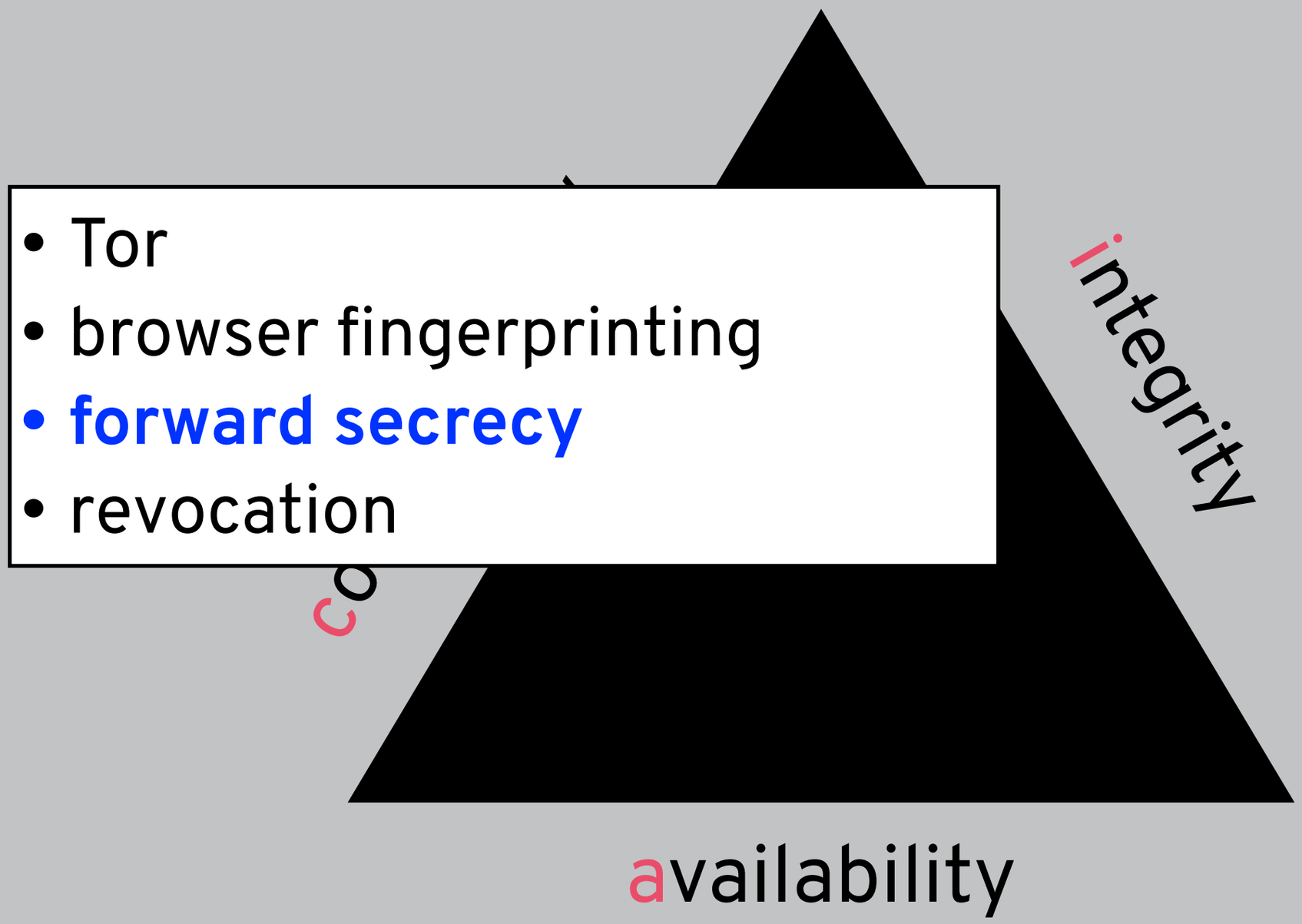
Is the server trusted by the browser? or the user?

- Browser fingerprinting

BROWSER FINGERPRINTING DEMO

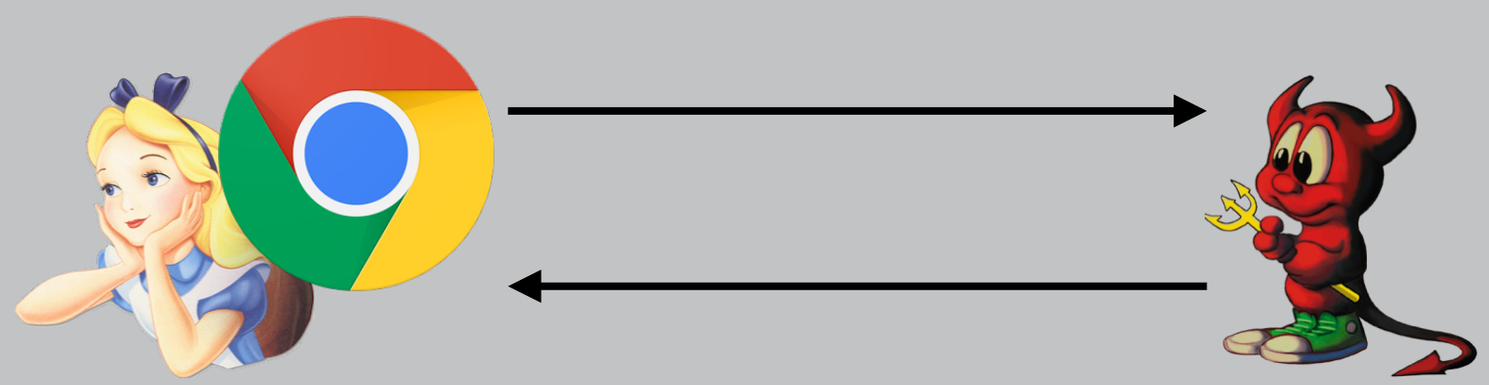
Your browser has a nearly-unique fingerprint

CONFIDENTIALITY, REVISITED



- Tor
- browser fingerprinting
- **forward secrecy**
- revocation

THREAT MODEL



Is the server trusted by the browser? or the user?

- Browser fingerprinting
- Forward secrecy

COMPROMISED CERTIFICATES

Public Key Info

Algorithm RSA Encryption (1.2.840.113549.1.1.1)

Parameters None

Public Key 256 bytes: AE 25 F8 F2 28 B4 61 93 4D 41 AA 75 5F
23 6F 17 6C 5C 11 3F 5B F3 1C 83 0B BE 6C C2 CD
C8 D4 BB 2A BF BD 1C 82 9C 5B 6B B5 1F ED 06 43
74 8F D3 B9 CE 0D 52 95 D0 61 C8 A0 8B 68 C0 CE
10 C2 C4 2D B4 45 A4 CB C9 E5 A0 A9 5B 01 95 1F
12 0D 78 D7
BF C1 4A C4
F5 B2 44 EC
D6 AC 21 A2
3F 0A 3F EE
B6 E4 85 9F
C7 EA C5 EE
46 D4 75 65
37 DA 6F 76
42 66 32 CB
9A A4 04 AA
48 8F 1B

Exponent 65537

Key Size 2,048 bits

Key Usage Encrypt, Veri

public key pk, company knows corresponding secret key sk

Signature 256 bytes: 7D 27 FF F3 16 E0 0C 27 FD 35 76 01 BA
00 C6 BE 5C 33 65 E3 2E 3E AA 13 00 99 64 25 D5
DB BF 52 48 01 1B 69 E4 65 5E 62 33 A9 F7 36 49
FD 15 06 3C A7 C2 49 9B AF EE F7 9A 74 13 15 F9
F5 75 BA 56 F0 37 0D 73 49 F1 E4 6B 33 0C 5E 84
DE 69 00 3E 93 35 20 A7 28 D7 3E 4A A8 E1 41 F9
84 75 E7 A7 A6 CB 56 48 5C 8D 2A 5F D5 DF C3 9D
12 56 EA 4A 71 C2 FB 9C 1C C1 98 D6 BC 32 7F 2E
F6 7A 87 AD D4 7D B2 C7 F3 A9 45 B4 D7 7B C4 32
AB 7D C
49 C2 D
B5 8F C
9C 43 C

what if attacker learns sk?

Fingerprints

SHA-256 90 9E 42 E3 FF 35 8C 03 0E FB 0E 1F CB 3D 8A 1F
DA 8E 52 EB F9 0B 12 D3 8A 3C A8 D9 EE 14 AF 25

SHA-1 27 DA 3A F2 0C 25 C6 8B D1 3E 36 82 90 C2 8A 42
7B 42 34 94

COMPROMISED COMMUNICATION

step 1: agree on **cipher suite**

step 2: validate certificate

check H(certificate) = fingerprint

so adversary can read traffic in this session

check Verify(pk_{CA}, sig, $pk_{service}$)

step 3: establish session key

client sends c=Enc($pk_{service}$, sk)

service uses sk=Dec($sk_{service}$, c)

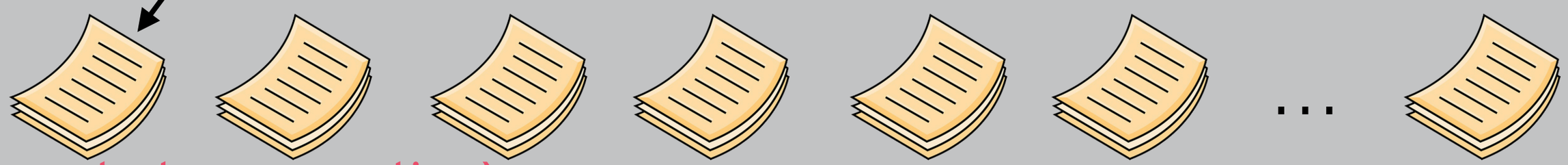
step 4: use sk to do AEAD

step 5: terminate connection (FIN)

COMPROMISED TRANSCRIPTS

service uses sk=Dec($sk_{service}$,c)

step 4: use sk to do AEAD



(encrypted conversation)

can the adversary now read all these conversations?!

FORWARD SECRECY

no! with **forward secrecy**, compromise of long-term keys doesn't affect previous sessions

use **ephemeral** values to establish sk

check Verify(pk_{CA}, sig, $pk_{service}$)

~~step 3: establish session key~~

~~client sends c=Enc($pk_{service}$, sk)~~

~~service uses sk=Dec($sk_{service}$, c)~~

step 4: use sk to do AEAD

FORWARD SECRECY

no! with **forward secrecy**, compromise of long-term keys doesn't affect previous sessions

"long-lived signing keys, short-lived encryption keys"

check Verify(pk_{CA}, sig, $pk_{service}$)

step 3: establish session key

client sends c=Enc($pk_{service}$, sk)

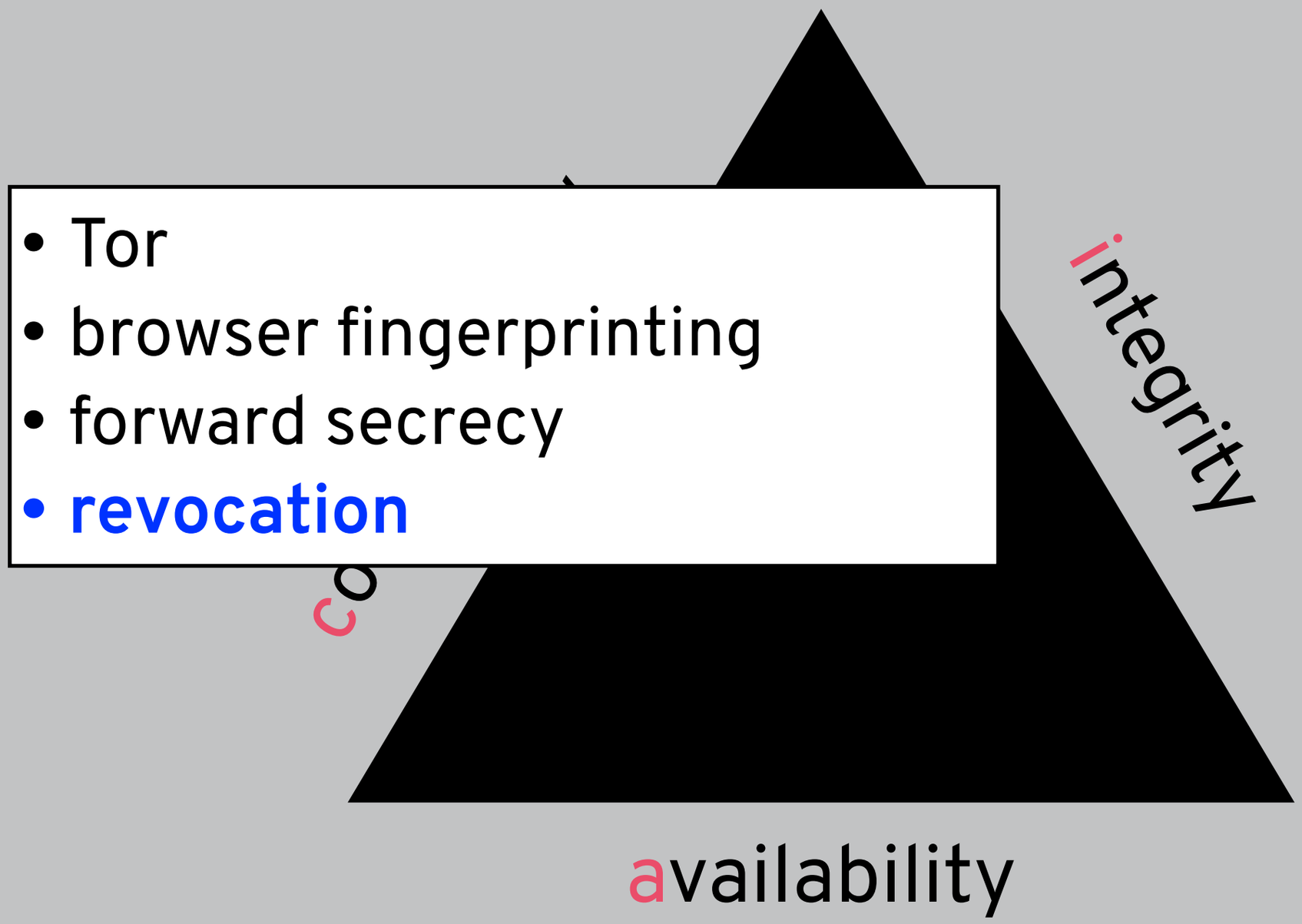
service uses sk=Dec($sk_{service}$, c)

step 4: use sk to do AEAD

FORWARD SECRECY

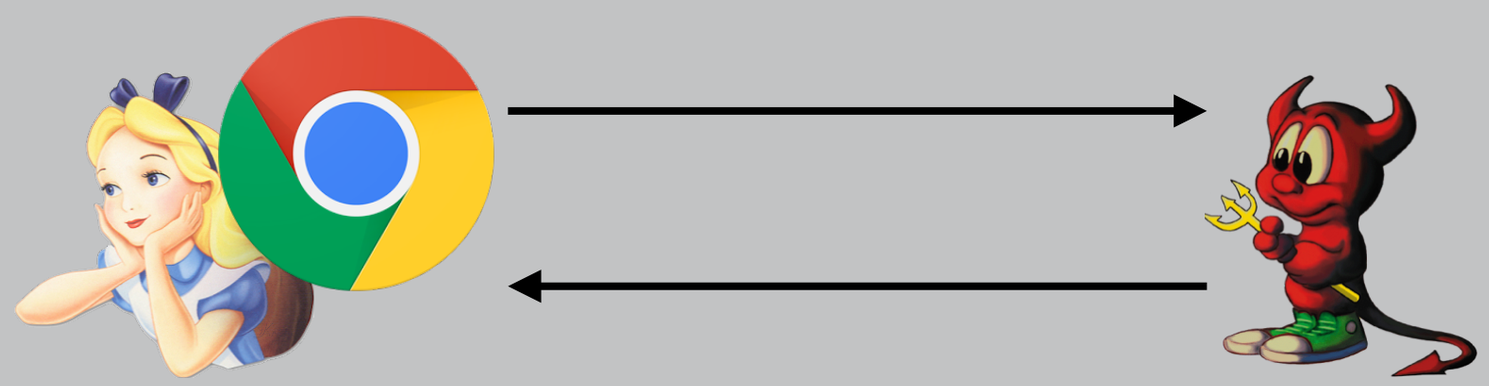
	Encrypted in transit?	Encrypted so the provider can't read it?	Can you verify contacts' identities?	Are past comms secure if your keys are stolen?
FACEBOOK	yes	no	no	no
IMESSAGE	yes	yes	no	yes
SIGNAL	yes	yes	yes	yes
TELEGRAM	yes	no	no	no
WHATSAPP	yes	yes	yes	yes

CONFIDENTIALITY, REVISITED



- Tor
- browser fingerprinting
- forward secrecy
- **revocation**

THREAT MODEL



Is the server trusted by the browser? or the user?

- Browser fingerprinting
- Forward secrecy / revocation

COMPROMISED CERTIFICATES

Public Key Info

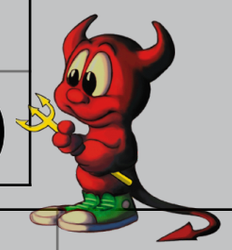
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	None
Public Key	256 bytes: AE 25 F8 F2 28 B4 61 93 4D 41 AA 75 5F 23 6F 17 6C 5C 11 3F 5B F3 1C 83 0B BE 6C C2 CD C8 D4 BB 2A BF BD 1C 82 9C 5B 6B B5 1F ED 06 43 74 8F D3 B9 CE 0D 52 95 D0 61 C8 A0 8B 68 C0 CE 10 C2 C4 2D B4 45 A4 CB C9 E5 A0 A9 5B 01 95 1F 12 0D 78 D7 26 E2 0B F8 F3 A6 A5 38 C3 61 F0 58 BF C1 4A C4 51 95 3E 78 40 C1 5A CD 2A C3 5C 9C F5 B2 44 EC 27 13 98 F7 7F 48 B0 02 23 95 93 1B D6 AC 21 A2 5A AD 64 2F E9 48 EF FC 92 81 16 B1 3F 0A 3F EE C8 D4 7E C1 0C 5F CF 06 80 09 9A 76 B6 E4 85 9F 17 4 C7 EA C5 EE A9 46 D4 75 65 74 37 DA 6F 76 EA 42 66 32 CB DE 42 1D 1B F3 7D B8 D6 C0 5B FB 79 9A A4 04 AA A2 0F 66 20 9D 76 0E 8E A3 F5 D2 DA 48 8F 1B
Exponent	65537
Key Size	2,048 bits
Key Usage	Encrypt, Verify, Wrap, Derive

public key pk, company knows corresponding secret key sk

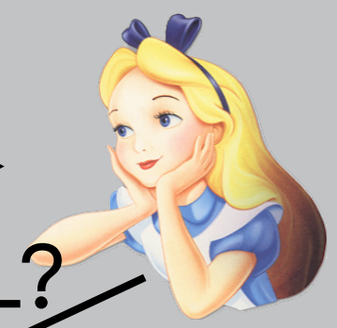
attacker learns sk. now what?

CERTIFICATE REVOCATION LISTS

alice.com	pk_A	$Sign(sk_{CA},pk_A)$
eve.com	pk_E	$Sign(sk_{CA},pk_E)$
…	…	…
bob.com	pk_B	$Sign(sk_{CA},pk_B)$



$cert_B$



in CRL?

…	…	…
…	…	…
bob.com	$cert_B$	$Sign(sk_{CA},cert_B)$

certificate revocation list (CRL)

(pk_{CA},sk_{CA})

add when certificate:

- is compromised
- was issued by mistake
- belongs to a defunct service

LIMITATIONS OF CRLS

how often to update CRL? how long to store?

places even more trust in CAs (decide when to revoke)

DoS attack on PKI shuts down certificate acceptance

scalability? how big can these things grow?

Online Certificate Status Protocol (OCSP)
addresses some of these but has its own tradeoffs

CONFIDENTIALITY, REVISITED