The Crypto Wars

Why is Cryptography a Problem?

- Strong cryptography helps protect secrets—yours, mine, businesses', the government's, and more
- Modern cryptography, if properly used, is essentially impossible to break
- But—SIGINT (signals intelligence, which includes cryptanalysis) is very important
- Historians estimate that the British SIGINT effort at Bletchley Park and the American analog in the Pacific shortened World War II by two years
- These days, breaking through strong encryption is very important to law enforcment, too
- Where is the balance?

A Brief History

- Cryptography has been with us for thousands of years
- ☞ The ancient Greeks and Romans employed encryption; even today, we speak of a *Caesar cipher*
- In Victorian times, lovers would sometimes communicate via encrypted messages in newspaper “Personals” columns
- (For amusement, Babbage and Wheatstone used to cryptanalyze them)

Definitions

[Cryptography]

Cryptanalysis breaking codes and ciphers; ability to read traffic without knowing the key

Cryptography "secret writing"; creating codes and ciphers, and using them

- **Codes** operate on semantic concepts, i.e., words, phrases, or sentences; they're seldom used today.
- **Ciphers** operate syntactically, i.e., on letters or bits, without regard to meaning.

Cryptology The academic field, including cryptography and cryptanalysis

An Excerpt From a 1926 Police Code

PERSONAL AND PHYSICAL DESCRIPTION.

atu	stature, height
atx	tall (over 171 cm = over 5ft. 7½)
aty	short (under 158 cm = under 5ft 2)
atz	taller, the taller
axa	shorter, the shorter
axb	very tall (177 to 188 cm—5ft. 10 to 6ft. 2.
axd	the tallest
axe	very short
axf	the shortest
axg	strikingly tall
axh	strikingly short
axi	medium height (between 159 and 170 cm = 5ft 2½ and 5ft 7)
axk	over medium height
axl	under medium height
axm	rather tall
axo	rather short
axp	dwarfish (under 4ft 9in.)

ayu	about 166 cm = 5ft 5½
ayx	about 167 cm = 5ft 5¾
ayy	about 168 cm = 5ft 6
ayz	about 169 cm = 5ft 6½
aza	about 170 cm = 5ft 7
azb	about 171 cm = 5ft 7½
azd	about 172 cm = 5ft 7¾
aze	about 173 cm = 5ft 8
azf	about 174 cm = 5ft 8½
azg	about 175 cm = 5ft 9
axh	about 176 cm = 5ft 9½
azi	about 177 cm = 5ft 9¾
azk	about 178 cm = 5ft 10
azl	about 179 cm = 5ft 10¼
azm	about 180 cm = 5ft 10½
azo	about 181 cm = 5ft 11
azp	about 182 cm = 5ft 11½
azr	about 183 cm = 6ft
azs	about 184 cm = 6ft 0½
azt	about 185 cm = 6ft 1
azu	about 186 cm = 6ft 1½
azx	about 187 cm = 6ft 1¾

60-Second Cryptology Tutorial

- A cipher is a pair of mathematical functions:

$$
\begin{aligned}
C &\leftarrow F(K, P) \\
P &\leftarrow F'(K, C)
\end{aligned}
$$

  that use a *key* to map *plaintext* to *ciphertext* and ciphertext to plaintext
- A key is a large, random number
- If you know the key, you can convert ciphertext to plaintext
- If you don't, it should be impossible to invert the function
- It's always conceptually possible to try every possible key, so your design should have far more keys than can be tried
- Always assume that your enemy knows F and F'

The Caesar Cipher

- Represent each letter by a number: $A \rightarrow 0, B \rightarrow 1, \ldots, Z \rightarrow 25$
- The key K is a number in $[1, 25]$
- Encryption: $C = (P + K) \bmod 26$
- (N.B. the Roman alphabet had 23 letters, not 26. . .)
- In English: replace each letter with the one K further down in the alphabet, wrapping around if necessary
- Obviously very weak: only 25 possible keys; just try them all

☞ The number of possible keys is an *upper bound*—not a lower bound—on the strength of an encryption algorithm

Public Key Cryptography

- Decryption may use a different key K' not derivable from K; this is called public key cryptography, because encryption key K can be public
- But K is derived from K'
- Public key crypto is at the heart of all Internet encryption
- Invented by Cocks and Ellis at GCHQ (the British equivalent to the NSA) in 1970; they called it *non-secret encryption*
- Reinvented in the open sector by Diffie and Hellman in 1975; Ralph Merkle had some of the concepts, too
- The best-known public key algorithm, RSA, was invented at MIT by Rivest, Shamir, and Adleman

Cryptanalysis is Traditional

- Clear, sophisticated descriptions of cryptanalysis in a 9th century Arabic book (and possibly a lost 8th century Arabic book), implying a long history of practicing it.
- During the Renaissance, major European governments had “Black Chambers”—organizations that would intercept diplomats’ mail, cryptanalyze them, and reseal the messages with forged seals
- (This implies that countries also had people devising codes)
- King Philip II of Spain complained to the Vatican that King Henry IV of France must be using black magic to read his codes, since there was no other way they could be broken

☞ The pope did nothing, since his own Black Chamber had also broken the Spanish codes, without resorting to the supernatural. . .

Spying on Communications Remained Important

- Britain was the hub of the 19th century international telegraph network
- They used this to intercept other countries' messages
- Their own messages went via the "all red route": telegraph lines that only came ashore somewhere in the British Empire

The American Black Chamber

- The US originally did little of this, but learned rapidly during World War I
- (During the Civil War, the Union was fairly competent at cryptology but the South was *really* bad)
- Herbert Yardley, with money from the State Department and the War Department, created the post-World War I *American Black Chamber*; among other things, it spied on delegates' traffic during the 1921 Naval Disarmament Conference, with particular focus on Japan
- In 1929, new Secretary of State Henry Stimson declared "Gentlemen do not read each other's mail" and shut down the operation
- (Other countries didn't have the same attitude. . .)
- The military continued its cryptanalytic activities, including (during the 1930s) a focus on Japanese diplomatic traffic

Enter the Computer

- The NSA has *always* used computers for cryptology
- The Army and Navy used standard and customized punch card equipment for cryptanalysis, starting around 1930
- During the 1950s and 1960s, the NSA was a major force behind the development of high-end computer technology
- Probably by 1965, they started using using computers to do cryptography (my opinion)
- Started developing bit-oriented ciphers no later than that (ditto)
- The NSA evaluated—and helped with and tampered with—the development of the Data Encryption Standard in 1976
- In the late 1970s, they started developing computer security standards

DES—The Data Encryption Standard

- In 1974, the National Bureau of Standards (NBS—now NIST, the National Institute of Standards and Technology) issued a public call for a cipher to protect unclassified communications
- IBM responded with “Lucifer”, a cipher with 112-bit keys (i.e., 2^{112} possible keys)
- The NSA evaluated it
- The eventual design had 56-bit keys
- Why? From a (redacted) declassified NSA history: “NSA worked closely with IBM to strengthen the algorithm against all except brute force attacks and to strengthen substitution tables, called S-boxes. Conversely, NSA tried to convince IBM to reduce the length of the key from 64 to 48 bits. Ultimately, they compromised on a 56-bit key.” (https://cryptome.org/0001/nsa-meyer.htm)

The War on Crypto: 1970s

- The NSA wanted the National Science Foundation to stop funding crypto research
- The NSA put secrecy orders on crypto patent applications
- On his own time, NSA employee Joseph Meyer sent the IEEE a letter warning that publishing crypto papers without prior government approval might be in violation of the International Traffic in Arms Regulations (ITAR): cryptography is a *munition*
- The NSA was upset that public key encryption had been reinvented in the open community—they still considered it secret
- They contemplated pushing for legislation restricting crypto research publication
- NSA director Bobby Inman set up a voluntary review process for academic papers; it died for lack of participation

Why?

- The NSA really wanted to protect Americans' data
- But they also wanted to spy on everyone else's
- If they controlled cryptography, they thought that they could control who used it
- Remember: this is pre-Internet

The 1980s—A Quiet Time

- There weren't many civilian users of cryptography, so the NSA didn't have to worry much
- The Diffie-Hellman (4,200,770) and RSA patents issued (4,405,829)
- (The Internet didn't start opening up until the end of the decade)
- But—during this period, cryptography became a serious academic discipline
- Cryptographers learned enough to build serious systems
- Ironically enough, DES was the foundation for this work
- The war would resume soon enough

The 1990s—The War Resumes

- Secure email
- Secure web
- DES is due for a replacement

PGP—Pretty Good Privacy

- In 1991, Phil Zimmermann develops and releases PGP, a secure email program.
- ☞ For this, he is investigated by the FBI for violating ITAR
- He's also hassled by the patent owner for violating the RSA patents

Export-Grade Crypto

- SSL appears in the first commercial web browser
- In the US, one could use 1024-bit public keys and a 128-bit symmetric key
- The export version, though, used 512-bit public keys and 40-bit symmetric keys
- Arithmetic: at 1 μ-sec/guess and 1,000 computers guessing, the solution to a 40-bit cipher will pop out in $<$ 20 minutes—trivial for a major intelligence agency
- But—trivial for *any* major intelligence agency, not just the NSA

AT&T's Telephone Security Device

Photo courtesy Matt Blaze

- AT&T built a simple-to-use telephone encryption device that used DES
- The FBI and the NSA were scared—bad guys would buy them and have strong protection against wiretaps
- (Could the NSA have cracked DES? Probably, but 2^{56} is still expensive and they wouldn't want their ability introduced in court.)

The Clipper Chip

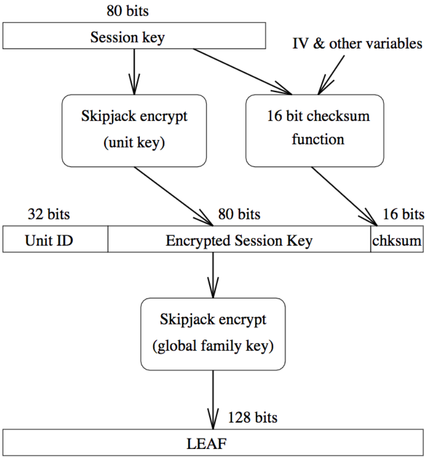
Photo courtesy Matt Blaze

- The NSA persuaded AT&T to replace DES with the *Clipper Chip*
- The Clipper chip used a classified 80-bit cipher, *Skipjack*, which implemented *key escrow*
- It met the NSA's goals—but it had to be done in *hardware*

Key Escrow

- The escrow agent—generally the government—has a copy of an additional decryption key
- With Clipper, there was a 128-bit LEAF (Law Enforcement Access Field): a copy of the *session key* encrypted with the *unit key*, plus the chip serial number, all encrypted with the family key
- The unit key was split into two parts, each held by a separate escrow agent

Structure of the LEAF



(Diagram courtesy Matt Blaze)

In Other Words. . .

- Skipjack was a strong cipher, and $2^{24}\times$ stronger than DES against a brute force attack
- The system was readable, but only by US agencies: the *NOBUS* (Nobody But Us) property
- It was intended to balance security and national security needs

Key Escrow Went Nowhere

- The private sector didn't want it
- They didn't want an extra chip, they didn't trust the security of the whole escrow system, and their customers didn't see the need
- Matt Blaze found a way to use the Clipper chip without a proper LEAF, i.e., with no key escrow (the "LEAFBLOWER" attack)
- No one outside the US was even vaguely interested—and a large percentage of sales by big US companies was to non-US customers

Deep Crack

- For years, outsiders had warned that DES was vulnerable to brute force attacks
- The NSA had always denied this, even (at times) with misleading responses
- But if it was vulnerable in 1979, surely Moore's Law would have made it more so by the 1990s
- The EFF funded creation of "Deep Crack", an open source hardware design for a DES-cracking engine, and proved that it worked
- It only cost them $250,000

The End of the Crypto Wars?

- American industry wasn't buying into key escrow
- There was a strong need to do crypto in software
- Non-US companies were taking advantage of export controls to grab market share
- The need for ubiquitous, strong crypto was becoming increasingly clear, but export controls were discouraging vendors from implementing crypto; they wanted one code base and worldwide interoperability
- The NSA gave up and declassified Skipjack
- The US government gave up on Clipper and (mostly) abolished export controls on mass-market crypto

A New Millenium?

- In an open, worldwide, competition, NIST standardized a design by two Europeans as AES (the Advanced Encryption Standard). AES can take 128-bit, 192-bit, and 256-bit keys.
- Although the NSA did help with the evaluation of the candidate algoriths, the open community broadly agreed with NIST's choice of finalists
- The NSA has stated that 256-bit AES is—when properly implemented and used—suitable for Top Secret traffic
- All seemed well, but the FBI and the NSA were still worried about "going dark"...

Going Dark

- Since 2011, the FBI has been warning Congress that it is “going dark”
- They’ve complained about the lack of lawful intercept as well as about encryption: “although the government may obtain a court order authorizing the collection of certain communications, it often serves that order on a provider who does not have an obligation under CALEA”
- But: what about all of the new metadata? What about the FBI’s ability to hack into computers?
- And the NSA wasn’t doing nothing. . .

Random Number Generator Standards

- Recall that cryptographic keys are supposed to be random numbers
- For complicated reasons, random numbers are used in other ways in cryptography; some of these values are transmitted unencrypted
- Computers are bad at randomness, so they use *pseudo-random generators* with a true-random *seed*:

```
function random() {
    static S
    S := F(S)
    return G(S)
}
```

- *G* should not be invertible, or an attacker who sees the output of the function would be able to recover *S* and all future values, e.g., keys

DUAL_EC_DRBG

- When NIST was standardizing some pseudo-random number generators, the NSA said "use this one": DUAL_EC_DRBG
- NIST was puzzled; it seemed very slow
- The NSA said "trust us; it's necessary for national security"—but didn't say why. . .
- NIST figured it was harmless to include: it was so slow that no one would use it
- Allegedly, though, the NSA paid RSA Data Security to make it the default in their popular BSAFE package
- BSAFE is heavily used for cryptography in embedded systems, including on-board encryptors for network cards

How It Worked

- DUAL_EC_DRBG relied on some arbitrary constants P and Q
- If you know a d such that $Q = d \times P$, you can invert invert the function
- Or, you can *select* a random d and use it to create Q
- In that case, you can invert G and predict all future outputs—typically, keys—after you've seen a few

A Clever, NOBUS Design

- An invertible PRNG would be too dangerous; anyone else could read traffic
- DUAL_EC_DRBG is more clever than that: it's effectively a public key encryption system, and only the NSA knows the decryption key K' (which is d)
- Result: the NSA can invert G; no one else can
- The possibilty was detected by outsiders, but it didn't draw much attention until the Snowden revelations confirmed it

It’s Not Just Communications

- Local police are concerned with encrypted devices
- The NSA is interested in communications
- The FBI does both

Apple iPhone Encryption

- *All* content is encrypted with one of a set of randomly-generated AES keys
- These keys are themselves protected: either encrypted with a key derived from a random UID that is stored in a secure, on-chip area (in newer iPhones), or encrypted with a key derived from the UID and the PIN
- (One of the cryptographic operations involved in the derivation is applied iteratively, to slow down guessing attempts.)
- Hardware-enforced maximum guess rate of 80 ms/try—because of a high iteration count, it takes that long to convert a PIN into a key

The FBI versus iOS 8

- Assume custom software, either provided by Apple or via an FBI jail-break of the phone
- Try all possible PINs:

4 digits	800 seconds
6 digits	22 hours
6 lower-case letters or digits	5.5 years
6 letters or digits	144 years
8 arbitrary characters	253,678 centuries

☞ These numbers assume *random* PINs. Is that assumption valid? Almost certainly not: https://freedom-to-tinker.com/blog/jbonneau/guessing-passwords-with-apples-full-device-encryption/

- Also: after 10 failed guesses, the phone erases all keys

The San Bernadino Shooters Case

- Syed Farook had a county-owned iPhone 5C
- It was last backed up to iCloud six weeks before the shootings
- There was some chance—though by all accounts, not much—that there would be some relevant information on the phone
- The phone was running iOS 9, i.e., a version that encrypts sensitive storage

The Court Order

- At the FBI's request, a magistrate judge ordered Apple to produce software that would allow unlimited tries
- Apple resisted; a hearing was scheduled
- Apple estimated it would take 3-10 person-months to produce the necessary code
- (My own, independent estimate was 4-6 person-months, but I forgot about managers and documentation.)
- There were technical flaws in Apple's protection scheme, making the unlock possible. (Apple has fixed these flaws. . .)
- In a similar case in Brooklyn—but involving iOS 7, where Apple did have the ability to unlock the phone—a different magistrate judge ruled against the FBI

Problem Solved?

- The FBI and Apple agreed: the encryption was airtight, unless Apple produced a custom version of the OS
- A vendor thought otherwise. . .
- The FBI paid an Australian firm $900,000 to crack the phone—and it worked
- Problem solved, case over?

The Tip of the Iceberg

- It wasn't just about just these one or two phones
- The FBI now has at least 8,000 phones they want to open
- The Manhattan DA's office has more than 3,800 such phones
- Other jurisdictions undoubtedly have many more
- Many commentators felt that the FBI was using the San Bernadino case—one where public sympathy and most of the facts favored them—to establish a legal precedent
- The FBI would really like a Federal law, but will settle for a court ruling if they have to
- And it may be a moot point—modern phones have enough bugs that private-sector software (MDFT—mobile device forensic tools) can unlock any phone

Warrant-Proof Encryption?

- Law enforcement has denounced “warrant-proof encryption”
- “Warrant-proof encryption defeats the constitutional balance by elevating privacy above public safety. Encrypted communications that cannot be intercepted and locked devices that cannot be opened are law-free zones that permit criminals and terrorists to operate without detection by police and without accountability by judges and juries.” (Deputy AG Rosenstein)

Is the FBI Right? Why Do Many Oppose Them?

- Technical and policy reasons
- Technical: Cryptography is *very* hard as is; complicating it leads to insecurity
- Policy: Who has access?
- Policy: Should life be easy for law enforcement?

Cryptography is Very Hard

- Devising correct cryptographic protocols is very hard
- (Remember LEAFBLOWER? Even the NSA got it wrong.)
- Implementing them is even harder
- 80% of mobile apps get *simple* crypto wrong
- Adding more complexity will lead to many more errors
- (At least one attempt to add key escrow to PGP resulted in insecurity: `https://www.cert.org/historical/advisories/CA-2000-18.cfm?`)
- It's much harder to protect communications than devices with key escrow because the attacker can observe and perhaps interfere with the encryption

We're Still Paying for the Export Controls

- The changes added to SSL to support export controls were *broken*
- New vulnerabilities directly traceable to those changes are still being found
- Cryptography is *hard*

Who Has Access?

- Which governments' requests should Apple honor?
- Right now, they don't have the code; it's easy for them to say "no" to all comers
- If they had built the code for the San Bernadino case, it would be trivial to modify it for other requests
- If some other government presents a communications intercept, is it from their own domestic criminal traffic? Or is it a foreign intelligence intercept? It's easier if they present a device.
- (But was the device seized at some border from a US traveler?)

Efficient Policing

- The Fourth Amendment was not intended to guarantee police access; rather, it was a *limit* on police powers
- Too much efficiency in policing is bad. In a Supreme Court case, Justice Sotomayor wrote of 'the ordinary checks that constrain abusive law enforcement practices: "limited police resources and community hostility".'
- Without unlocked phones or key escrow, police will have to work harder—but is that a bad thing?

The Fourth Amendment

"The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures, shall not be violated, and no warrants shall issue, but upon probable cause, supported by oath or affirmation, and particularly describing the place to be searched, and the persons or things to be seized."

On the Other Hand

- Sometimes, the best or only evidence will be protected by crypto
- Should we give up the ability to break the encryption?
- The NSA generally has to cope with adversaries who don't listen to US policies anyway
- ☞ A former NSA director has called this "the golden age of SIGINT"
- But even they'd benefit; many bad guys just use off-the-shelf products
- How should a decision be made?

Alternatives?

- Finding other evidence?
- Lawful hacking?
- More research and technical assistance?
- Remember the MDFTs!

Other Evidence?

- There is no reliable data on how often inability to break encryption has ruined a case
- Probably, there are some such cases—but how many?
- A Russian spy used very strong encryption—but the password was written on a yellow stickie under her keyboard
- The plural of anecdote is not data!

Lawful Hacking

- Perhaps the FBI can hack into locked devices
- It worked in the San Bernadino case, and many companies claim that they can (currently?) get into any iPhone
- Too expensive? Remember Justice Sotomayor's comment about limited police resources being an advantage.

More Technical Assistance

- The FBI has good technical abilities. Most local police departments do not.
- There are already some assistance programs—these could be stepped up
- MDFTs are off-the-shelf
- Local police need more training in how to deal with all sorts of computer-related crime

Client-Side Scanning

- Various bodies in the EU want devices to scan files for improper content: child sexual abuse material (CSAM), terrorist material, etc.
- Apple announced a concrete plan for doing such scanning on iOS 15, though they've backed off a bit and haven't shipped it yet
- Many issues!

Apple's Scheme

- Only scan material being shared via iCloud
- Use a *perceptual hash*—a function that tries to capture the essence of the image, despite transformations like cropping, resizing, etc.
- (There is an independent feature that attempts to use machine learning to detect when minors are trying to sext.)
- Match those hashes against CSAM databases from two different countries
- Use clever cryptography so that the images are only decryptable if at least 30 such images are shared
- Do not report anything until after manual inspection of the flagged content

Many Issues!

- Will the scanning be restricted to CSAM, or will some governments insist on adding other material?
- ☞ Terrorist material? Anti-government material? Hate speech?
- How will people know?
- It's easy for folks with real CSAM to bypass the filters—they're client-side, and hence the perceptual hash can be (and has been) reverse-engineered
- Can attackers install their own filters on the device?
- More suitable for bulk surveillance than narrowly targeted inspection with a warrant

Today's Status

- Right now, there is no U.S. requirement for exceptional access. Other countries would like it, too, but the big vendors have resisted
- The FBI keeps pushing
- Given today's mess in Washington, it's hard to predict what will happen—exceptional access is not a traditional partisan issue; the libertarian right and the civil libertarian left are allies on this issue
- Stay tuned!

Questions?

(Great blue heron, Morningside Park, December 24, 2019)