CSE107: Intro to Modern Cryptography

https://cseweb.ucsd.edu/classes/sp22/cse107-a/

Emmanuel Thomé

May 31, 2022

Lecture 18a

A few things about public-key cryptanalysis

Brute force

$O(\sqrt{\#G})$-time discrete logarithms

Subgroup attacks

Subexponential discrete logarithms and factoring

Caveat: breakable v. secure

We discuss several aspects of cryptanalysis and things that can be done.

When X can be broken in practice today, it is certainly not secure.

When X is hard to break in practice today, it DOES NOT MEAN that it is secure, because

- we should think of state-level adversarires;
- the security parameters must provide for decades of future use;
- we want some margin.

Plan

Brute force

$O(\sqrt{\#G})$-time discrete logarithms

Subgroup attacks

Subexponential discrete logarithms and factoring

What can be brute-forced?

It is always possible to attempt a break with brute force.

- DES, with key space 2^{55}, can be brute forced.
  (Smarter techniques exist, but in the end brute forcing works just as well in practice.)
- Anything that can be broken by the exploration of 2^{40} to 2^{60} possibilities is eminently a target for brute force:
  - If 2^{40}, it is a computation that requires very little resources, and can be done repeatedly.
  - On the other end of this spectrum, and on to 2^{80}-time computations, this becomes a significant cryptanalytic effort, which might make sense for high-value targets only.

Reducing to a large enumeration

Sometimes, cryptanalysis consists of a phase which reduces the problem to a large enumeration, which can then be done in a massively parallel way. This is not exactly brute force, but it does share some aspects with it.

- Collisions on SHA-1 are like that.

Thresholds

We want to say that X is secure given that it requires an enumeration of at least 2^k possibilities.

What is a good value of k? (see also note 214 on Piazza)

- Easily breakable: below $k \leq 60$.
- Doable with considerable effort or by state-level adversaries: $k \approx 80$.
- Safe now and for some time ahead: $k \geq 128$.

Note: the first two thresholds increase slightly over time.

Plan

Brute force

$O(\sqrt{\#G})$-time discrete logarithms

Subgroup attacks

Subexponential discrete logarithms and factoring

The discrete logarithm problem

Definition (The Discrete Logarithm Problem (DLP))

Let G be a group and $g \in G$. Let $h \in \langle g \rangle$. The discrete logarithm of h to base g is the integer k such that $g^k = h$. It is uniquely defined modulo the order of g.

In what follows, we let $m = \#\langle g \rangle$ be the order of g.

Note: in the definition, if we require that g be a generator of G, then m is also the order of G.

Baby-step Giant-step algorithm

There is an algorithm that computes discrete logarithms in any group.

Goal:

- Write the unknown discrete logarithm k as

$$k = iM + j$$

  for an integer $M \approx \sqrt{m}$ so that i and j are roughly the same size.
- Try to solve for i and j by finding matching values.

Important observation:

$$h = g^k \quad \Leftrightarrow \quad h = g^{iM} \cdot g^j$$
$$h(g^{-M})^i = g^j.$$

Baby-step Giant-step algorithm

Let $h = g^x$. Write $x = iM + j$ for a chosen integer M, with $0 \leq i \leq m/M$ and $0 \leq j \leq M$.

Goal: find i and j such that $h(g^{-M})^i = g^j$.

Algorithm:

- compute $\gamma = g^{-M}$.
- Baby steps: compute $S = \{g^j \mid 0 \leq j \leq M\}$.
- Giant steps: for $0 \leq i \leq m/M$, compute $h\gamma^i$ and stop if it is in S.

Complexity: $O(\sqrt{m})$ (deterministic, proven).

- if M is chosen to be $\lceil \sqrt{m} \rceil$
- if the test "is in S" is done in $O(1)$ (e.g., with hash tables)

Time and Space

Complexity of Baby-step Giant-step, as described.

- Time $O(\sqrt{m})$
- Memory $O(\sqrt{m})$

Improvements (probabilistic / heuristic):

- Time $O(\sqrt{m})$
- Memory $O(1)$

If this is all we can do. . .

The bound $O(\sqrt{m})$ is a universal upper bound on the hardness of cryptanalysis. It can never be harder. But it may be faster!

If it were a lower bound we would know how to parameterize public-key crypto quite well.

- Example: in order to provide 128-bit security (2^{128} operations needed), elliptic curves are chosen over 256-bit fields (leading to groups with $\approx 2^{256}$ elements).
- This is because for EC, no known algorithm is faster than $O(\sqrt{m})$.

Generic groups

An often-cited result:

Theorem (DLP in generic groups; Nechaev–Shoup)

In a generic group G, the cost of computing a DLP is asymptotically proportional to $\sqrt{\#G}$.

A generic group is a group about which we know NOTHING.

Problem: we always know something!
This result is important, but it has no practical impact.

Plan

Brute force

$O(\sqrt{\#G})$-time discrete logarithms

Subgroup attacks

Subexponential discrete logarithms and factoring

Groups and subgroups

Fact: If G is a cyclic group and $n = \#G$, then for each divisor d of n, there is a unique subgroup G_d of G such that $\#G_d = d$.

Proof: Let g be a generator. Pick $G_d = \langle g^{n/d} \rangle$.

Let us write $n = \prod_{i=1}^{r} p_i^{e_i}$.

Is it possible to use this for the computation of discrete logarithms in G?

Plan

Subgroup attacks

- CRT
- CRT and DLP
- Consequences of Pohlig-Hellman

The Chinese Remainder Theorem

Alice and Bob run around Crypto Wood, whose perimeter is one mile. (they do multiple laps)

- Alice runs a mile in 7 minutes.
- Bob runs a mile in 9 minutes.

I'm standing exactly where Alice and Bob started some time ago.

Alice finished a lap 3 minutes ago. Bob finished a lap 1 minute ago.

How long have they been running?

The Chinese Remainder Theorem

Let x = time since Alice and Bob started. We have:

$$\begin{cases} x \equiv 3 \pmod 7 \\ x \equiv 1 \pmod 9 \end{cases}$$

(note: it is important that 7 and 9 are coprime)

The Chinese Remainder Theorem

Let $x =$ time since Alice and Bob started. We have:

$$\begin{cases} x \equiv 3 \pmod 7 \\ x \equiv 1 \pmod 9 \end{cases}$$

(note: it is important that 7 and 9 are coprime)

Answer: $x = 10$. Or possibly $x = 73$, or $x = 136$, $\ldots$.

The answer is defined modulo $\operatorname{lcm}(7, 9) = 7 \times 9 = 63$.

The abstract point of view

The CRT is often written as:

Theorem

Let $n = \prod_{i=1}^{r} p_i^{e_i}$ *with all primes* p_i *distinct. We have an explicit ring isomorphism:*

$$\mathbb{Z}/n\mathbb{Z} \cong \mathbb{Z}/p_1^{e_1}\mathbb{Z} \times \cdots \times \mathbb{Z}/p_r^{e_r}\mathbb{Z}.$$

(this implies in particular the formula for $\varphi(n)$)

In practice, this statement says little about how we really use the CRT.

A CRT example

$$\begin{cases} x \equiv 3 \pmod{7} \\ x \equiv 1 \pmod{9} \\ x \equiv 8 \pmod{13} \end{cases}$$

Note: 7, 9, and 13 are pairwise coprime.

Algorithm:

- Compute $n = 7 \times 9 \times 13 = 819$.
- Compute
  - $n_7 = \frac{n}{7} = 117$ and $\lambda_7 = (n_7)^{-1} \bmod 7 = 3$.
  - $n_9 = \frac{n}{9} = 91$ and $\lambda_9 = (n_9)^{-1} \bmod 9 = 1$.
  - $n_{13} = \frac{n}{13} = 63$ and $\lambda_{13} = (n_{13})^{-1} \bmod 13 = 6$.
- Compute $x = 3n_7\lambda_7 + 1n_9\lambda_9 + 8n_{13}\lambda_{13} \bmod n =$ $(3 \times 117 \times 3) + (1 \times 91 \times 1) + (8 \times 63 \times 6) \bmod n = 73$.

Plan

CRT and DLP

Let $G = \langle g \rangle$. Assume $\#\langle g \rangle = m = \prod_{i=1}^{r} p_i^{e_i}$.

We want to find $\mathrm{DLog}_{G,g}(h) = k$, which is defined modulo $m = \#\langle g \rangle$.

Can we turn this into a CRT-like system of equations? YES.

Write $q_i = p_i^{e_i}$.

- G has a unique subgroup G_i of order q_i: $\langle g_i = g^{n/q_i} \rangle$.
- $h_i = h^{n/q_i}$ is also in G_i. Let $k_i = \mathrm{DLog}_{G_i,g_i}(h_i)$. We have:

$$h^{n/q_i} = g^{n/q_i \cdot k} = g_i^k = h_i = g_i^{k_i}.$$

- Therefore, if k_i is known, we know that $k \equiv k_i \pmod{q_i}$.

CRT/DLP example

Let $p = 6553$ and $g = 29$. The subgroup $G = \langle g \rangle$ of $\mathbb{Z}_p^*$ has order 819. Let $h = 6161$.

We want to find $\mathrm{DLog}_{G,g}(h) = k$, which is defined modulo $m = 819$.

- The subgroup G_1 of order 7 is generated by $g_1 = g^{117} = 4662$. We have $h^{117} = 3858$.

  In $G_1 = \langle g^{117} \rangle$:

i	0	1	2	3	4	5	6
g_1^i	1	4662	4496	3858	4564	6330	2301

  Thus $k_1 = \mathrm{DLog}_{G_1,g_1}(h_1) = 3$ and $k \equiv 3 \pmod 7$.
- In $G_2 = g^{91}$: $k_2 = \mathrm{DLog}_{G_2,g_2}(h_2) = 1$ and $k \equiv 1 \pmod 9$.
- In $G_3 = g^{63}$: $k_3 = \mathrm{DLog}_{G_3,g_3}(h_3) = 8$ and $k \equiv 8 \pmod{13}$.

Conclusion: $\mathrm{DLog}_{G,g}(h) = 73$ (as in previous example).

Everything happens in the subgroups

The most important aspects:

- The computation is broken into small pieces;
- everything happens in the subgroups;
- and in this example the subgroups are small.

Pohlig-Hellman, a.k.a. CRT for DLP

Theorem (Pohlig-Hellman reduction)

If $\#\langle g\rangle = \prod_{i=1}^{r} q_i$, then one way to solve the DLP in $\langle g\rangle$ is to look at all subgroups one after another, which costs at most:

$$O(\sqrt{q_1}) + \cdots + O(\sqrt{q_r}).$$

Corollary: if all q_i are small, then DLP is NOT HARD.
We need at least a large prime-order subgroup in order to have security.

Plan

Subgroup attacks

- CRT
- CRT and DLP
- Consequences of Pohlig-Hellman

Consequences of Pohlig-Hellman

Consequence #1:

- it is best to work in a well-chosen prime order subgroup
  - because it's no weaker,
  - and is computationally cheaper.

Consequences of Pohlig-Hellman

Consequence #1:

- it is best to work in a well-chosen prime order subgroup
    - because it's no weaker,
    - and is computationally cheaper.

Consequence #2:

- it is best to work in a well-chosen prime order subgroup
    - because doing so is less error-prone, and provides better security.

Examples of secret leak via subgroups

Simple example: $\mathbb{Z}_p^*$, g a generator of order $p-1$.

If $S = g^s$ is public data (e.g., it is a public key), then this leaks $s \bmod 2$:

- If $S^{(p-1)/2} = 1$, then s is even.
- If $S^{(p-1)/2} = -1 \bmod p$, then s is odd.

Examples of secret leak via subgroups

Worse: $\mathbb{Z}_p^*$, g a generator of order $p-1$.

- Size of p: 1024 bits, with primes of 1, 12, 47, 52, 414, 498 bits.
- Secret: a random 128-bit integer.
- Public key: g^s.

This is TOTALLY INSECURE!

Examples of secret leak via subgroups

Worse: $\mathbb{Z}_p^*$, g a generator of order $p-1$.

- Size of p: 1024 bits, with primes of 1, 12, 47, 52, 414, 498 bits.
- Secret: a random 128-bit integer.
- Public key: g^s.

This is TOTALLY INSECURE!

- Raise to the appropriate power to solve a DLP in a 12-bit prime-order subgroup.
- Do the same in the 47-bit and 52-bit prime-order subgroups. At worst, this is $\approx 2^{26}$ computations.
- We have found s modulo a $1 + 23 + 47 + 52 = 123$-bit number. Brute-force the rest.

Sadly, this is a real story!

Plan

Brute force

$O(\sqrt{\#G})$-time discrete logarithms

Subgroup attacks

Subexponential discrete logarithms and factoring

Groups with easier DLP

There are groups with easier DLP:

- Some even have completely trivial DLP, and are of course not used in crypto ($\mathbb{Z}_n$ with addition, for example).
- More interestingly, the DLP in multiplicative subgroups of finite fields can be computed with the Number Field Sieve algorithm.

From lectures 10-11

Computation	**Time**
DLP in $\mathbb{Z}_p^*$	$e^{1.92(\ln p)^{1/3}(\ln \ln p)^{2/3}}$ (roughly) subexponential time
Factorization of N	$e^{1.92(\ln N)^{1/3}(\ln \ln N)^{2/3}}$ (roughly) subexponential time

TL;DR: DLP modulo a k-bit prime and factoring a k-bit integer cost roughly the same.

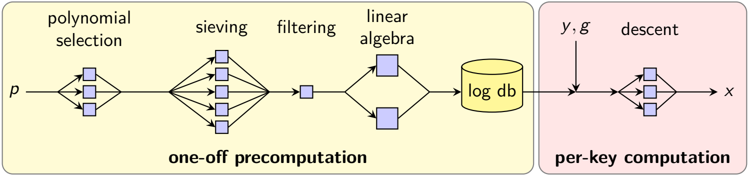
DH versus RSA

DLP underpins Diffie-Hellman. Factoring understands RSA.

- RSA: Each user has their own key. Factoring one key does not make it any easier to break another, similar size key.
- DH: It's different. p is typically a public, fixed parameter. A "key" is ONE challenge of the form $\mathrm{DLog}_{\mathbb{Z}_p^*,g}(y)$.

Precomputation for NFS DLP

Computation of $\mathrm{DLog}_{\mathbb{Z}_p^*,g}(y)$ with NFS goes like this:



Typical data (elapsed time using many machines):

	precomputation	per-key
Logjam (512 bits)	week	minutes
DLP-240 (795 bits)	months	hours

A different perspective

DLP in $\mathbb{Z}_p^*$: the per-key cost, while still subexponential, is several order of magnitude easier than the one-off precomputation.

A few fixed, very widespread primes used for DH could be high value targets for a massive DLP precomputation, which would make it possible to break many challenges (= many DH key exchanges) efficiently.

Take home messages

- Brute-forcing anything that requires 2^{60} computations or less is eminently doable, and cheap (at least around 2^{40}).
- In any group of size m, computing discrete logarithms takes at most time $O(\sqrt{m})$.
- When there are subgroups, the security is that of the largest prime-order subgroup. Computations should take place only in that subgroup.
- Some groups have much easier DL, and multiplicative subgroups of finite fields are among them. No fast DL is known for elliptic curves.
- Factoring and discrete logs in finite fields have similar hardness, BUT there is a huge difference in the per-key cryptanalysis cost.
- Cryptanalysis of soon-to-be-standardized PQ primitives keep trickling, and that should be a real concern.

CSE107: Intro to Modern Cryptography

https://cseweb.ucsd.edu/classes/sp22/cse107-a/

Emmanuel Thomé

May 31, 2022

Lecture 18b

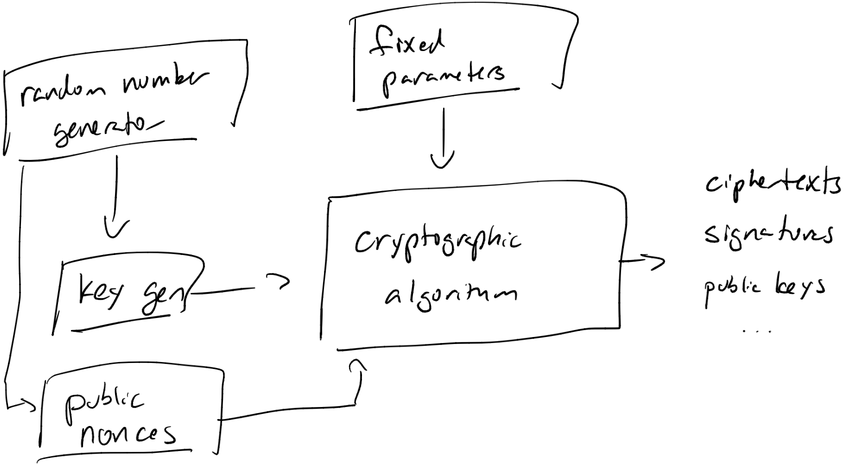
A History of Cryptographic Backdoors

Subverting cryptography

Plan

Subverting cryptography

Cryptographic Algorithm Components



If you wanted to subvert a cryptographic algorithm, how would you do it?

Suggested methods to subvert cryptography

- Design algorithm so that true key strength is less than apparent key strength.
- Choose "fixed" parameters to weaken algorithm strength.
- Choose "fixed" parameters to encode a secret.
- Weaken key generation algorithm to generate keys with less entropy.
- Use a flawed random number generator so that secrets are easier to predict.
- ...

Black Chamber: Forerunner of the NSA

Founded post WWI.

Closed down in 1929.

Henry L. Stimson:

"Gentlemen do not read each other's mail."

Crypto AG

Swiss company founded after WWII by Boris Hagelin.

1950s–1960s: Company paid by CIA to weaken algorithms.

1970: Bought in secret by CIA and German BND.

Machines used by dozens of countries from 1950s to 2000s.

Employees: “The algorithms always looked fishy.” “Not all questions appeared to be welcome.”

WaPo: “the secret partners adopted a set of principles for rigged algorithms... They had to be ‘undetectable by usual statistical tests’ and, if discovered, be ‘easily masked as implementation or human errors.”’

Decades of rumors confirmed in 2019.

https://www.washingtonpost.com/graphics/2020/world/national-security/cia-crypto-encryption-machines-espionage/

Late 1970s: DES

NSA made two changes to IBM's algorithm:

- Changed key strength from 64 to 56 bits: overt weakening.
- Changed S-boxes. Suspected to be a backdoor but later discovered to protect against differential cryptanalysis.

The "crypto wars" in the US

- Crypto wars 1.0
  - Late 1970s,
  - US government threatened legal sanctions on researchers who published papers about cryptography.
  - Threats to retroactively classify cryptography research.
- Crypto wars 2.0
  - 1990s
  - Main isssues: Export control and key escrow
  - Several legal challenges
- Crypto wars 3.0
  - Now
  - Snowden
  - Apple v. FBI
  - ...?
  - Calls for "balance"

TBC

Continued on Thursday