3010 Lecture Week 12

Key Management, Key agreement & Loose ends

Dr Tay Kian Boon

RSA Factoring Record

- RSA 829, with hard Number Field Sieve Algorithms, computation time

- 2700 core-years, 2GHz PC!

How to Agree on Secret Key without meeting Each Other -Sounds Impossible?

Possible!

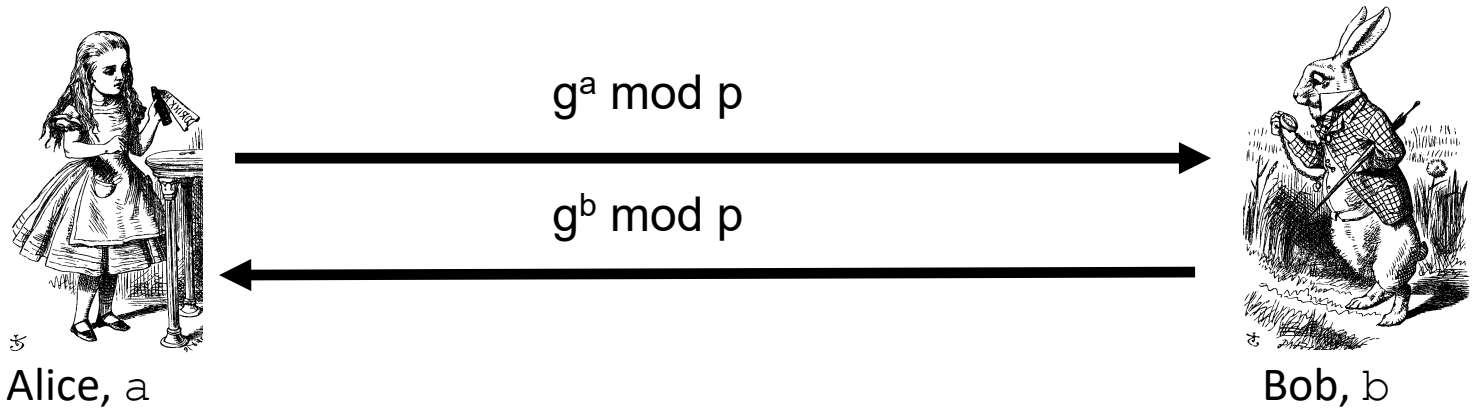
Diffie & Helmman (1977)

Diffie-Hellman Key Exchange

- Let p be prime, let g be a **generator**
  - i.e. For any $x \in \{1,2,\ldots,p\text{-}1\}$ there is n s.t. $x = g^n \bmod p$
- Alice selects randomly her private value a
- Bob selects randomly his private value b
- Alice sends $g^a \bmod p$ to Bob
- Bob sends $g^b \bmod p$ to Alice
- Both compute shared secret, $g^{ab} \bmod p$
- Shared secret can be used as symmetric key!

Diffie-Hellman Key Exchange

- **Public:** g and p
- **Private:** Alice's exponent a, Bob's exponent b



- Alice computes $(g^b)^a = g^{ba} = g^{ab} \bmod p$
- Bob computes $(g^a)^b = g^{ab} \bmod p$
- They can use $K = g^{ab} \bmod p$ as symmetric key

Diffie-Hellman Key Exchange - Remarks

- Suppose Bob and Alice use Diffie-Hellman to determine symmetric key $K = g^{ab} \bmod p$
- Eavesdropper Eve can see $g^a \bmod p$ and $g^b \bmod p$
  - But… $g^a g^b \bmod p = g^{a+b} \bmod p \neq g^{ab} \bmod p$
- If Eve can find a or b, she gets K
  - **Hard problem: discrete log** problem (subexponential complexity)

Diffie-Hellman Key Exchange - Remarks

- Diffie-Hellman key exchange is one of the most useful algorithms for key exchange

- We shall see an example in next slide.

Diffie-Hellman Key Exchange - Example

$$p = 2\,147\,483\,659,\ q = 2\,402\,107,\ \text{and}\ g = 509\,190\,093,$$

but in real life one would take $p \approx 2^{2048}$. Note that g has prime order q in the field $\mathbb{F}_p$. The following diagram indicates a possible message flow for the Diffie–Hellman protocol:

Alice		Bob
$a = 12\,345$		$b = 654\,323,$
$\mathfrak{el}_A = g^a = 382\,909\,757$	$\longrightarrow$	$\mathfrak{el}_A = 382\,909\,757,$
$\mathfrak{el}_B = 1\,190\,416\,419$	$\longleftarrow$	$\mathfrak{el}_B = g^b = 1\,190\,416\,419.$

The shared secret group element is then computed via

$$\mathfrak{el}_A{}^b = 382\,909\,757^{654\,323} \pmod p = 881\,311\,606,$$

$$\mathfrak{el}_B{}^a = 1\,190\,416\,419^{12\,345} \pmod p = 881\,311\,606,$$

with the actual secret key being given by $k = H(881\,311\,606)$ for some KDF H, which we model as a random oracle.

Diffie-Hellman Key Exchange – Present Record

- On 2 Dec 2019, Fabrice Boudot, Pierrick Gaudry, Auror Guillevic, Nadia Heninger, Emmanuel Thomé, and Paul Zimmermann announced the computation of a discrete logarithm modulo the 240-digit (795 bit prime)
- This computation was performed using very complicated Number Field Sieve (NFS) algorithms and the open-source CADO-NFS software.
- Want to find out how long it took?

Diffie-Hellman Key Exchange – Present Record

- Using Intel Xeon Gold 6130 CPUs as a reference (2.1GHz), the discrete logarithm alone part of the computation took approximately

- **3100 core-years!**

Diffie-Hellman Key Exchange – Secure Parameters (2023)

- Prime p at least 1024 bit – (my prediction -falling in the next couple of years)
- For longer term security, use 1536 or 2048-bit prime p
- Make sure the special number (p-1) has a super large prime factor (of size almost size of prime p)
- One way to ensure this is to choose prime p such that p-1 = 2q, where q is prime.

SOME KEY MANAGEMENT ISSUES

KEY MANAGEMENT INTRODUCTION-OWASP

- This Key Management Cheat Sheet provides developers with guidance for implementation of cryptographic key management within an application in a secure manner.
- It is important to document rules and practices for:
  - key life cycle management (generation, distribution, destruction)
  - key compromise, recovery and “zeroization” (“SECURE WIPE”)
  - key storage (will cover this)
    - The rest, not tested: https://cheatsheetseries.owasp.org/cheatsheets/Key_Management_Cheat_Sheet.html
  - key agreement

KEY STORAGE (OWASP)-Focus on RED

- Developers must understand where cryptographic keys are stored within the application. Understand what memory devices the keys are stored on.
- Keys must be protected on both volatile and persistent memory, ideally processed within secure cryptographic modules.
- Keys should never be stored in plaintext format.
- Ensure all keys are stored in cryptographic vault, such as a hardware security module (HSM) or isolated cryptographic service.
- If you are planning on storing keys in offline devices/databases, then encrypt the keys using Key Encryption Keys (KEKs) prior to the export of the key material. KEK length (and algorithm) should be equivalent to or greater in strength than the keys being protected.

KEY STORAGE

- Ensure that keys have integrity protections applied while in storage (consider dual purpose algorithms that support encryption and Message Code Authentication (MAC)).
- Ensure that standard application level code never reads or uses cryptographic keys in any way and use key management libraries.
- Ensure that keys and cryptographic operation is done inside the sealed vault SUCH AS HSM
- All work should be done in the vault (such as key access, encryption, decryption, signing, etc).

THE END!