

1. Let \mathbb{G} be a finite group such that all elements $y \in \mathbb{G}$ can be expressed as powers of $g \in \mathbb{G}$. Then the discrete logarithm problem is following. Given $y \in \mathbb{G}$, find a smallest integer x such that $g^x = y$ in finite group \mathbb{G} . Discrete logarithm problem is known to be hard in general, i.e., all universal algorithms for computing logarithm run in time $\Omega(\sqrt{|\mathbb{G}|})$.
 - (a) Show that for a fixed group \mathbb{G} , there exists a Turing machine that finds the discrete logarithm for every $y \in \mathbb{G}$ in $O(\log_2 |\mathbb{G}|)$ steps.
 - (b) Show that for a fixed group \mathbb{G} , there exists an analogous Random Access Machine that achieves the same efficiency.
 - (c) Generalise the previous construction and show that for every fixed function $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$ there exists a Turing machine and a Random Access Machine such that they compute $f(x)$ for every input $x \in \{0, 1\}^n$ in $O(n + m)$ steps.
 - (d) Are these constructions also valid in practise? Explain why these inconsistencies disappear when we formalise algorithms through universal computing devices.

Hint: What is the time-complexity of binary search algorithms?

2. Consider a classical Turing machine without internal working tapes, i.e., the Turing machine has a single one-sided (input) tape that initially contains inputs and must contain the desired output after the execution.
 - (a) Show that all sorting algorithms take at least $\Omega(n^2)$ steps where n is the total length of inputs x_1, \dots, x_k . What is the time-complexity of best sorting algorithms? Explain this contradiction.
 - (b) Does the minimal time-complexity change if the Turing machine has internal working tapes?
 - (c) Sketch how one can simulate execution of Random Access Machines on a Turing machine. What is the corresponding overhead?
 - (★) Construct a set of tasks that can be implemented significantly more efficiently on Turing machines with $\ell+1$ working tapes than on Turing machines with ℓ tapes.

Hint: It is well-known fact that reversing n -bit string takes $\Omega(n^2)$ steps on a Turing machine without working tapes.

3. Let $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_5$ be algorithms for finding discrete logarithm such that the success probability is bounded

$$49 \cdot \text{Adv}_{\mathbb{G}}^{\text{dl}}(\mathcal{A}_i) \geq \Pr [x \leftarrow \mathcal{A}_i(y) : y = g^x] \geq 7 \cdot \text{Adv}_{\mathbb{G}}^{\text{dl}}(\mathcal{A}_i)$$

if $\pi(y) = 1$. Find tight lower and upper bounds of $\text{Adv}_{\mathbb{G}}^{\text{dl}}(\mathcal{A})$ for the following adversary \mathcal{B}

$$\mathcal{B}(y) \begin{cases} i \leftarrow_{\mathcal{U}} \{1, 2, 3\}, x \leftarrow \mathcal{A}_i(y) \\ \text{if } \pi_i(y) = 1 \text{ then} \\ \quad \begin{cases} \text{if } g^x \neq y \wedge \pi_4(y) = 1 \text{ then return } \mathcal{A}_4(y) \\ \text{else return } x \end{cases} \\ \text{else if } \pi_5(y) = 1 \text{ then return } \mathcal{A}_5(y) \\ \text{else return } \mathcal{A}_1(y) \end{cases}$$

provided that $\Pr[y \leftarrow_{\mathcal{U}} \mathbb{G} : \pi_i(y) = 1] = \frac{1}{42+i}$ and $\text{Adv}_{\mathbb{G}}^{\text{dl}}(\mathcal{A}_i) = i^2 \cdot \varepsilon$.

4. Let \mathbb{G} be a finite q -element group such that all elements $y \in \mathbb{G}$ can be expressed as powers of $g \in \mathbb{G}$.
 - (a) Let \mathcal{A} be an algorithm that always finds a discrete logarithm with the expected running-time τ . Construct a t -time algorithm \mathcal{B} that fails with probability 2^{-80} and its running-time t is linear in τ .
 - (b) Let \mathcal{A} be an algorithm for finding the highest bit of discrete logarithm such that $\Pr[\mathcal{A}(y) \text{ guesses correctly}] \geq \varepsilon > \frac{1}{2}$ for any $y \in \mathbb{G}$. Construct an algorithm that fails with probability 2^{-80} .
 - (c) Let \mathcal{A} be a discrete logarithm finder that uses algorithm \mathcal{A} five times to get inputs for the aggregating algorithm \mathcal{C}

$$\mathcal{B}(y) \begin{cases} x_1 \leftarrow \mathcal{A}(y), \dots, x_5 \leftarrow \mathcal{A}(y) \\ \text{return } \mathcal{C}_1(x_1, \dots, x_5) \end{cases}$$

The construction guarantees that \mathcal{C} succeeds in finding the discrete logarithm of y if all x_i are correct. Find the $\text{Adv}_{\mathbb{G}}^{\text{dl}}(\mathcal{B})$ if

$$\Pr[y \leftarrow \mathbb{G} : \text{the output of } \mathcal{A}(y) \text{ is correct}] = \varepsilon.$$

Hints: Use Chebyshev's, Jensen's and Markov's inequalities.

5. A cryptosystem is a triple of algorithms $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ such that the equality $\mathcal{D}(\mathcal{E}(m, k), k) = m$ holds for all messages $m \in \mathcal{M}$ and keys $k \leftarrow \mathcal{K}$. Cryptosystem is perfectly secure if a ciphertext c reveals nothing about the corresponding message m , i.e., $\Pr[m|c] = \Pr[m]$.
 - (a) Prove that cryptosystem is perfectly secure only if $H(m|c) = H(m)$. What about the implication to the other direction?
 - (b) Show that $H(k, m, c) \geq H(m|c) + H(c)$. For which enciphering algorithms does the equality $H(k, m, c) = H(m|c) + H(c)$ hold?

- (c) Show that $H(k, m, c) = H(k) + H(c|k)$. Conclude that cryptosystem is perfectly secure only if $H(k) \geq H(m)$.
- (d) Show that $H(k|c) = H(m) + H(k) + H(c|m, k) - H(c)$. What does the result mean in practise?
6. Estimate how much time is needed to break the following three file encryption methods without using cipher-specific attacks.
- (a) The file is encrypted with 128-bit AES cipher and the key is stored in a special file that is protected with a password. Namely, the key is encrypted with another key that is derived from the password.
 - (b) The file is encrypted with old 56-bit DES cipher and the key is stored in the special file that is encrypted with a public key. The corresponding secret key is stored in the ID card.
 - (c) The file is encrypted with 80-bit IDEA cipher and the key is stored in the special file that is encrypted with a public key. The corresponding secret key is stored in the TPM chip.
7. There are many ways how to attack a standard e-banking system. First, an attacker can distribute malware that logs all kinds of passwords. Secondly, an attacker can send out forged e-mails that instruct bank customers to send passwords to a certain mail account. Thirdly, an attacker can attack the underlying cryptographic protection mechanism. When the attacker has a control over the account, he or she has to withdraw the money through an auxiliary account belonging to a mule. This poses a risk as mules do not always deliver the money to attacker's account.

Compute a success probabilities of all attack scenarios and find the one with highest expected gain, given only some estimates of conditional probabilities. Namely, let **Malware**, **Phishing** and **CryptoBreak** denote success in the first attack step. Let **Detect** denote the event that an unauthorised bank transfer or the attack itself is detected. Finally, let **Cheat** denote the event that mule cheats and the attacker does not get the money. Then

$$\begin{array}{ll}
 \Pr[\text{Malware}] = 10^{-3} & \Pr[\text{Detect}|\text{Malware}] = 10^{-4} \\
 \Pr[\text{Phishing}] = 10^{-2} & \Pr[\text{Detect}|\text{Phishing}] = 1 \\
 \Pr[\text{CryptoBreak}] = 10^{-27} & \Pr[\text{Detect}|\text{CryptoBreak}] = 0 \\
 \Pr[\text{Detect}|\text{Draw } 100] = 10^{-2} & \Pr[\text{Cheat}|\text{Draw } 100] = 0 \\
 \Pr[\text{Detect}|\text{Draw } 1000] = 10^{-1} & \Pr[\text{Cheat}|\text{Draw } 1000] = 10^{-1} \\
 \Pr[\text{Detect}|\text{Draw } 10000] = 1 & \Pr[\text{Cheat}|\text{Draw } 1000] = 10^{-2}
 \end{array}$$

What is probability that the corresponding attacks remain unnoticed?

- (□) Let \mathcal{A} be a solver for the Computational Diffie-Hellman problem with the advantage $\text{Adv}_{\mathbb{G}}^{\text{cdh}}(\mathcal{A}) = \varepsilon > \frac{1}{2}$. Now consider a success amplification by

majority voting

$$\mathcal{B}^{\mathcal{A}}(x, y) \left[\begin{array}{l} \text{For } i \in \{1, \dots, n\} \text{ do} \\ \quad \left[\begin{array}{l} a \xleftarrow{u} \mathbb{Z}_q, b \xleftarrow{u} \mathbb{Z}_q \\ z_i \leftarrow \mathcal{A}(xg^a, yg^b) \cdot x^{-b}y^{-a}g^{-ab} \end{array} \right. \\ \text{Output the most frequent value among } z_1, \dots, z_n. \end{array} \right.$$

Find a better lower bound of the advantage $\text{Adv}_{\mathbb{G}}^{\text{cdh}}(\mathcal{B})$ than was given in the lecture. Show that your bound is asymptotically tight.

- (★) Let \mathbb{G} be a finite q -element group such that all elements $y \in \mathbb{G}$ can be expressed as powers of $g \in \mathbb{G}$. Let $\psi : \mathbb{Z}_q \rightarrow \{0, 1\}$ be a non-trivial linear predicate, i.e., $\psi(x + y) = \psi(x) \oplus \psi(y)$ and $\psi(x) \neq 0$ for some x . Show that if there exists an efficient procedure \mathcal{A} with the advantage

$$\text{Adv}_{\mathbb{G}}^{\psi}(\mathcal{A}) = \Pr[y \leftarrow \mathbb{G} : \mathcal{A}(y) = \psi(\log y)] > \frac{1}{2} ,$$

then it possible to compute discrete logarithm efficiently, i.e., the running-time of the construction depends linearly on the running-time of \mathcal{A} for fixed advantage $\text{Adv}_{\mathbb{G}}^{\psi}(\mathcal{A})$. How does the running-time depend on $\text{Adv}_{\mathbb{G}}^{\psi}(\mathcal{A})$?

- (★) Let \mathbb{G} be a finite q -element group such that all elements $y \in \mathbb{G}$ can be expressed as powers of $g \in \mathbb{G}$. Show that if there exists an efficient procedure \mathcal{A} that can always compute the highest bit of $\log y$ then the discrete logarithm problem is easy. Extend the proof to the case where the advantage $\text{Adv}(\mathcal{A}) = \Pr[y \leftarrow \mathbb{G} : \mathcal{A}(y) \text{ guesses correctly}] > \frac{1}{2}$. How does the running-time depend on $\text{Adv}(\mathcal{A})$?