Crittografia 2
Prerequisiti

Matteo Rossi
Cybersecurity National Lab

https://cybersecnatlab.it

License & Disclaimer

License Information

Disclaimer

- We disclaim any warranties or representations as to the accuracy or completeness of this material.
- Materials are provided “as is” without warranty of any kind, either express or implied, including without limitation, warranties of merchantability, fitness for a particular purpose, and non-infringement.
- Under no circumstances shall we be liable for any loss, damage, liability or expense incurred or suffered which is claimed to have resulted from use of this material.

Argomenti

- Operazioni binarie (XOR)
- Aritmetica modulare
- La libreria "pycryptodome"

Operazioni binarie

- Le primitive (simmetriche) moderne sono costruite "a strati":
  - Ognuno di questi strati esegue solitamente una o più operazioni
  - Tutta la costruzione viene poi iterata più volte

Operazioni binarie

- Queste operazioni possono essere:
  - Operazioni non lineari (eventualmente molto complesse)
  - Tabelle di sostituzione (SBOX)
  - Operazioni "semplici" eseguite bit per bit
    - AND
    - OR
    - XOR
    - …

Operazioni binarie

- Queste operazioni possono essere:
  - Operazioni non lineari (eventualmente molto complesse)
  - Tabelle di sostituzione (SBOX)
  - Operazioni "semplici" eseguite bit per bit
    - AND
    - OR
    - XOR ← Focus di questa sezione
    - …

Exclusive-Or

L'*exclusive-or* (XOR) è un'operazione binaria indicata con $\oplus$, o con ^ in alcuni linguaggi di programmazione, con la seguente tabella di verità:

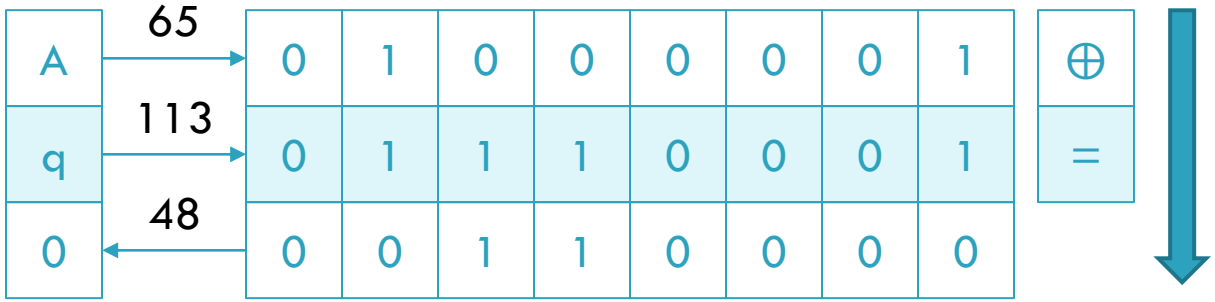
a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive-Or

- In pratica, per fare lo XOR di due caratteri:
  - Convertiamo i caratteri (ASCII) in binario
  - Facciamo lo XOR verticalmente
  - Riconvertiamo il risultato

Exclusive-Or

- Esempio: $A \oplus \text{"q"} = \text{"0"}$



Challenges

Crypto 04:

https://training.olicyber.it/challenges#challenge-329

Exclusive-Or

- Proprietà di base:
  - $a \oplus (b \oplus c) = (a \oplus b) \oplus c$
  - $a \oplus b = b \oplus a$
  - $a \oplus a = 0$
  - $a \oplus 0 = a$
  - $a \oplus b \oplus a = b$

One-Time Pad (Vernam, 1917)

- Idea: usiamo lo XOR direttamente per cifrare messaggi
  - $Enc(k, m) = k \oplus m = c$
  - $Dec(k, c) = k \oplus c = m$
  - Questo cifrario viene chiamato *"One-Time Pad"* (OTP)
  - Nota: la chiave k dev'essere generata in maniera casuale

One-Time Pad (Vernam, 1917)

- One-Time Pad è un cifrario "perfetto"
  - Perché? Intuitivamente: ogni bit del risultato può essere 0 o 1 con la stessa probabilità!

One-Time Pad (Vernam, 1917)

- One-Time Pad è un cifrario "perfetto"
  - Perché? Intuitivamente: ogni bit del risultato può essere 0 o 1 con la stessa probabilità!
- Assunzioni "scomode":
  - Dobbiamo avere una chiave lunga almeno quanto il testo

Challenges

Crypto 05:
https://training.olicyber.it/challenges#challenge-330

One-Time Pad (Vernam, 1917)

- One-Time Pad è un cifrario "perfetto"
  - Perché? Intuitivamente: ogni bit del risultato può essere 0 o 1 con la stessa probabilità!
- Assunzioni "scomode":
  - Dobbiamo avere una chiave lunga almeno quanto il testo
  - La chiave può essere utilizzata per una sola cifratura

Come si rompe "One-Time Pad"?

- Se la chiave viene utilizzata più volte:
  - Il primo carattere di ogni messaggio sarà XOR-ato sempre con lo stesso byte
  - Il secondo carattere di ogni messaggio sarà XOR-ato sempre con lo stesso byte
  - Il terzo carattere di ogni messaggio sarà XOR-ato sempre con lo stesso byte
  - … e così via

Come si rompe "One-Time Pad"?

- Idea: possiamo fare lo stesso ragionamento della challenge Crypto 05 ma "per colonne"
  - Proviamo tutti i 256 candidati per il primo byte
  - Controlliamo quando tutto è stampabile/rispetta una certa distribuzione (frequenze dei caratteri!)
  - Ordiniamo dal più probabile al meno probabile
  - … ripetiamo per tutti gli altri!

Come si rompe "One-Time Pad"?

- Fortunatamente qualcuno ha già scritto dei tool per farlo in maniera automatica al posto nostro:
  - https://github.com/CameronLonsdale/MTP
  - https://github.com/hellman/xortool

Challenges

Crypto 06:

https://training.olicyber.it/challenges#challenge-331

One-Time Pad (Vernam, 1917)

- Domanda: perché abbiamo scelto proprio lo XOR?
- Cosa succederebbe con AND oppure OR?
- Challenge (per casa): https://training.olicyber.it/challenges#challenge-81

Curiosità: esistono davvero cifrari perfetti?

Teorema (Shannon): per ottenere un cifrario perfetto servono più "possibili chiavi" che "possibili messaggi"

Provate a dimostrarlo!

Aritmetica modulare

- Classificazione (poco rigorosa) delle primitive:
  - Simmetriche: basate su algoritmi euristici
  - Asimmetriche: basate su problemi matematici "forti"

Aritmetica modulare

- Classificazione (poco rigorosa) delle primitive:
  - Simmetriche: basate su algoritmi euristici
  - Asimmetriche: basate su problemi matematici "forti"
    - Fattorizzazione
    - Operazioni su curve ellittiche
    - Reticoli
    - Codici a correzione d'errore
    - …

Aritmetica modulare

Per noi tutto ciò che è asimmetrico si baserà sull'aritmetica modulare, il cui concetto fondamentale è quello di *congruenza*

Congruenze

- Dati tre interi a, b, n diciamo che a è *congruo* a b *modulo* n ($a \equiv b \bmod n$) se (equivalentemente):
  - $a - b$ è divisibile per n
  - a e b danno lo stesso resto se divisi per n

Congruenze – Esempi

- $32 \equiv 7 \bmod 5$
  - $32 - 7 = 25$ divisibile per 5 ($25/5 = 5$ resto 0)
  - $32 / 5 = 6$ resto 2 e $7 / 5 = 1$ resto 2

Congruenze – Esempi

- $32 \equiv 7 \bmod 5$
  - $32 - 7 = 25$ divisibile per 5 ($25/5 = 5$ resto 0)
  - $32 / 5 = 6$ resto 2 e $7 / 5 = 1$ resto 2
- $91 \not\equiv 18 \bmod 3$
  - $91 - 18 = 73$ non divisibile per 3 ($73 / 3 = 24$ resto 1)
  - $91 / 3 = 30$ resto 1 e $18 / 3 = 6$ resto 0

Congruenze

- Proprietà:
  - $a \equiv a \bmod n$
  - $a \equiv b \bmod n \Rightarrow b \equiv a \bmod n$
  - $a \equiv b \bmod n$ e $b \equiv c \bmod n \Rightarrow a \equiv c \bmod n$

Challenges

Crypto 08:

https://training.olicyber.it/challenges#challenge-333

Congruenze

- Se $a \equiv a' \bmod n$ e $b \equiv b' \bmod n$ allora:
  - $a + b \equiv a' + b' \bmod n$
  - $ab \equiv a'b' \bmod n$
  - $ka \equiv ka' \bmod n$ per qualsiasi k intero
  - Nota: questo non vale per la divisione

Inverso moltiplicativo

- Idea intuitiva: vogliamo definire un oggetto che "si comporti" come una divisione
  - $\frac{1}{4} = 0.25 \ (mod\ 10)$ ha poco senso
  - Idea migliore: $a \cdot \frac{1}{a} = 1$ nei razionali, usiamo questa "definizione"
  - Vogliamo costruire un a^{-1} tale che $a \cdot a^{-1} \equiv 1 \ (mod\ n)$, e questo oggetto lo chiamiamo *inverso moltiplicativo*

Inverso moltiplicativo

- Identità di Bézout:
  - dati a, b interi allora esistono sempre x, y interi tali che

$$x \cdot a + y \cdot b = MCD(a, b)$$

Idea di "dimostrazione": algoritmo di Euclide esteso

Inverso moltiplicativo

- Applichiamolo ora a una coppia a, n con $MCD(a, n) = 1$, cosa succede?
  - Troviamo x, y tali che $a \cdot x + n \cdot y = 1$
  - $a \cdot x + n \cdot y \equiv 1 \ (mod\ n)$
  - $a \cdot x \equiv 1 \ (mod\ n)$
  - $x = a^{-1} (mod\ n)$

Inverso moltiplicativo

- Teorema di Bézout: dati a e n interi positivi
  - L'inverso moltiplicativo di $a \ (mod \ n)$ esiste se e solo se $\mathrm{MCD}(a, n) = 1$
  - Quando esiste, l'inverso è unico $(mod \ n)$

Challenges

Crypto 09:

https://training.olicyber.it/challenges#challenge-334

La libreria "pycryptodome"

- Nelle challenge non troverete (quasi mai) le primitive direttamente implementate, ma verranno usate delle librerie:
  - pycryptodome
  - cryptography
  - gmpy2
  - …

La libreria "pycryptodome"

- La libreria più utilizzata è "pycryptodome"
  - Sul portale è presente un "cheatsheet" sul suo utilizzo: https://training.olicyber.it/api/file/13563f96-8ffa-4a10-a60b-b2d1aa6f53a9/pycryptodome_basics.pdf
  - Sono presenti inoltre due challenge "tutorial" sulla libreria:
    - https://training.olicyber.it/challenges#challenge-332
    - https://training.olicyber.it/challenges#challenge-339

Crittografia 2
Prerequisiti

Matteo Rossi
Cybersecurity National Lab

https://cybersecnatlab.it