Digitalizzazione e principi fondamentali della crittografia dei dati

Prof. Loriano Storchi
loriano@storchi.org
https://www.storchi.org/

Rappresentazione binaria dei numeri (breve digressione)

- In un **sistema di numerazione posizionale,** data la **base** , questa definisce direttamente il numero di **simboli (cifre)** utilizzati per scrivere il numero.
  - Ad esempio, nel sistema decimale utilizziamo 10 simboli (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
- I moderni sistemi di numerazione sono posizionali, **quindi il numero viene scritto specificando l'ordine delle cifre e ogni cifra assume un valore a seconda della sua posizione.**
  - Ad esempio $423 = 4 * 10^{2} + 2 * 10^{1} + 3 * 10^{0}$
  - Se vuoi 4 centinaia, 2 decine e 3 unità

Rappresentazione binaria dei numeri (breve digressione)

- In generale, data una **base b, avrò b simboli** (cifre) e quindi **un intero N** sarà scritto come:
  - Valore $N = c_n * b^n + c_{n-1} * b^{n-1} + ... + c_0 * b^0$
- Allo stesso modo, se ho un numero $N = 0, c_1, c_2, ..., c_{\square}$
  - Valore $N = c_1 * b^{-1} + c_2 * b^{-2} + ... + c_n * b^{-n}$
- Consideriamo ora il caso più semplice, quello della rappresentazione degli **interi senza segno (i numeri naturali).**
- Se utilizzo un sistema di numerazione in **base b con n cifre,** potrò rappresentare al **massimo b^n numeri diversi** , quindi tutti i numeri da 0 ad $ab^n - 1$
- Ad esempio, in **base 10** è chiaro che usando **due cifre posso rappresentare tutti i numeri da 0 a 99,** quindi **$10^2 = 100$ numeri** distinti.

La base binaria

- Utilizzando una base 2, quindi solo due simboli 0 e 1, rimanendo sempre nell'ambito della rappresentazione degli interi positivi con n cifre, potrò rappresentare al massimo tutti i numeri compresi tra **0 e 2^n -1**
  - Ad esempio, utilizzando due cifre posso rappresentare 4 numeri distinti:
  - $00 = 0 * 2^1 + 0 * 2^0 = 0$
  - $01 = 0 * 2^1 + 1 * 2^0 = 1$
  - $10 = 1 * 2^1 + 0 * 2^0 = 2$
  - $11 = 1 * 2^1 + 1 * 2^0 = 3$

I pesi posizionali

Nel sistema binario, ogni posizione rappresenta una potenza di 2. Il numero si legge da destra a sinistra partendo da 2^0.

	2^4	2^3	2^2	2^1	2^0
	2	**2**	**2**	**2**	**2**
...	**4**	**3**	**2**	**1**	**0**

Valore	16	8	4	2	1
Binario					1

Esempio di conversione

1011

BASE 2 (BINARIO)

Il processo di addizione

Mappiamo il numero binario ai `1011`si posizionali e sommiamoli:

$$(1 \times 8) + (0 \times 4) + (1 \times 2) + (1 \times 1)$$

$$8 + 0 + 2 + 1 = 11$$

Da decimale a binario

Tradurre i numeri umani in logica computazionale utilizzando il **metodo della divisione per 2** .

Il metodo della divisione per 2

1. L'algoritmo

- ÷ Dividi il numero decimale di partenza per 2.
- Annota il **resto** (sarà sempre 0 o 1).
- Prendi il nuovo quoziente e dividilo di nuovo per 2.
- ↑ Ripeti l'operazione finché il quoziente non è 0.

Leggi i resti dal basso verso l'alto.

2. Esempio: Converti 13

```
13 ÷ 2 = 6   ...  Remainder 1 (LSD) ↑
 6 ÷ 2 = 3   ...  Remainder 0       |
 3 ÷ 2 = 1   ...  Remainder 1       |
 1 ÷ 2 = 0   ...  Remainder 1 (MSD) ↑

Result: 1101
```

Perché il binario è importante

Limitazioni hardware fisiche

I computer non comprendono i numeri astratti. A livello hardware fisico, una CPU è composta da miliardi di transistor microscopici.
Questi transistor hanno solo due stati fisici affidabili e misurabili per quanto riguarda l'elettricità:

- `1` = Alta tensione (interruttore acceso)
- `0` = Bassa tensione (interruttore spento)

Il sistema binario rappresenta il livello di astrazione più basso in assoluto. Tutto ciò che un computer fa, dalla visualizzazione di un'immagine all'esecuzione di un videogioco, viene in definitiva tradotto in sequenze di 1 e 0.

Il sistema esadecimale

Cos'è la base 16?

The Standard 10

Humans naturally count in **Decimal (Base-10)** because we have ten fingers.

This system uses exactly ten distinct symbols to represent all possible numbers before rolling over and adding a new column:

```
0, 1, 2, 3, 4, 5, 6, 7, 8, 9
```

The Extra 6

Hexadecimal (Base-16) requires sixteen distinct symbols.

Since we run out of standard numeric digits after 9, we borrow the first six letters of the English alphabet to represent values 10 through 15:

```
A, B, C, D, E, F
```

La magia del 16

16

VALUES PER DIGIT

The Perfect Binary Match

Why do computer scientists use Base-16 instead of Base-10? Because of its perfect mathematical alignment with binary logic.

$$16 = 2^4$$

This simple equation means that **one single hexadecimal digit perfectly represents exactly 4 bits** of binary data (known as a "nibble").

Consequently, it takes exactly **two** hex digits to represent a full 8-bit Byte.

La tabella di traduzione

Observe how a single Hex character maps flawlessly to a 4-bit binary sequence.

DECIMAL (BASE-10)	BINARY (BASE-2)	HEXADECIMAL (BASE-16)
0	0000	0
5	0101	5
10	1010	A
15	1111	F
255 (Full Byte)	1111 1111	FF

Applicazioni nel mondo reale

Web Colors

Digital colors mix Red, Green, and Blue light from 0 to 255. In CSS and design software, this is written as a 6-digit hex code.

`#FF0000` (Pure Red)

Memory & Pointers

When debugging software or viewing memory dumps, RAM addresses are always displayed in Hex to keep the output readable.

`0x7FFF5FBFF8A0`

Networking

Hardware MAC addresses and the new standard of IPv6 internet addresses use Hex to manage massive numeric identities.

`00:1A:2B:3C:4D:5E`

Compressione visiva dei dati

For the Human Eye.

Reading endless streams of raw 1s and 0s is virtually impossible for human programmers.

Hexadecimal acts as a visual compression algorithm. By grouping bits into blocks of 4, we reduce the physical length of binary strings by **75%**.

It does not change how the computer stores the data underneath; it simply changes how we view it.


COMPUTER SCIENCE
DIGITALIZZAZIONE
STATISTICS
MACHINE LEARNING

La base binaria

- Un byte (un frammento) rappresenta oggi la sequenza di 8 bit ed è storicamente diventato l'elemento base dell'indirizzabilità e quindi l'unità di misura fondamentale dell'informazione.
- **8 bit significa che con 1 byte posso rappresentare $2^8 = 256$ valori diversi** . Quindi, nel caso di numeri interi senza segno, i numeri da 0 a 255. **Se utilizzo un bit per indicare il segno, ad esempio 0 per positivo e 1 per negativo,** posso rappresentare gli interi da -128 a 127 (da 10000000 a 01111111).
- Oppure con 8 bit posso rappresentare 256 caratteri diversi.

ASCII

- ASCII esteso
- utilizzare 8 bit
- ASCII originale
- US-ASCII 7-bit

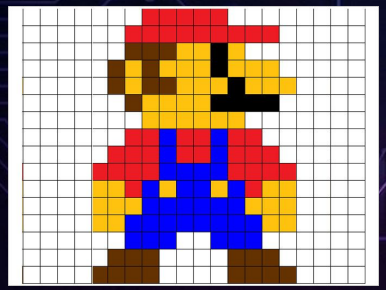
000	NUL	033	!	066	B	099	c	132	ä	165	Ñ	198	ã	231	þ
001	Start Of Header (SOH)	034	"	067	C	100	d	133	à	166	ª	199	Ã	232	Þ
002	Start Of Text (STX)	035	#	068	D	101	e	134	å	167	º	200	╚	233	Ú
003	End Of Text (ETX)	036	$	069	E	102	f	135	ç	168	¿	201	╔	234	Û
004	End Of Transmission (EOT)	037	%	070	F	103	g	136	ê	169	®	202	╩	235	Ù
005	Enquiry	038	&	071	G	104	h	137	ë	170	¬	203	╦	236	ý
006	Acknowledge (ACK)	039	'	072	H	105	i	138	è	171	½	204	╠	237	Ý
007	Bell	040	(	073	I	106	j	139	ï	172	¼	205	═	238	¯
008	Backspace (BS)	041	)	074	J	107	k	140	î	173	¡	206	╬	239	´
009	Horizontal Tab	042	*	075	K	108	l	141	ì	174	«	207	¤	240	
010	Line Feed **(LF)**	043	+	076	L	109	m	142	Ä	175	»	208	ð	241	±
011	Vertical Tab	044	,	077	M	110	n	143	Å	176	░	209	Ð	242	‗
012	Form Feed **(FF)**	045	-	078	N	111	o	144	É	177	▒	210	Ê	243	¾
013	Carriage Return **(CR)**	046	.	079	O	112	p	145	æ	178	▓	211	Ë	244	¶
014	Shift Out	047	/	080	P	113	q	146	Æ	179	│	212	È	245	§
015	Shift In	048	0	081	Q	114	r	147	ô	180	┤	213	ı	246	÷
016	Dataline Escape (DLE)	049	1	082	R	115	s	148	ö	181	Á	214	Í	247	¸
017	DC 1 (XON)	050	2	083	S	116	t	149	ò	182	Â	215	Î	248	°
018	DC 2	051	3	084	T	117	u	150	û	183	À	216	Ï	249	¨
019	DC 3 (XOFF)	052	4	085	U	118	v	151	ù	184	©	217	┘	250	·
020	DC 4	053	5	086	V	119	w	152	ÿ	185	╣	218	┌	251	¹
021	Negative Acknowledge (NAK)	054	6	087	W	120	x	153	Ö	186	║	219	█	252	³
022	Synchronous Idle	055	7	088	X	121	y	154	Ü	187	╗	220	▄	253	²
023	End Of Transmission Block	056	8	089	Y	122	z	155	ø	188	╝	221	¦	254	■
024	Cancel	057	9	090	Z	123	{	156	£	189	¢	222	Ì	255	
025	End Of Medium	058	:	091	[	124	|	157	Ø	190	¥	223	▀		
026	Substitude	059	;	092	\	125	}	158	×	191	┐	224	Ó		
027	Escape **(ESC)**	060	<	093	]	126	~	159	ƒ	192	└	225	ß		
028	File Seperator	061	=	094	^	127 (DEL)	⌂	160	á	193	┴	226	Ô		
029	Group Seperator	062	>	095	_	128	Ç	161	í	194	┬	227	Ò		
030	Record Seperator	063	?	096	`	129	ü	162	ó	195	├	228	õ		
031	Unit Seperator	064	@	097	a	130	é	163	ú	196	─	229	Õ		
032	*SPACE* (SP)	065	A	098	b	131	â	164	ñ	197	┼	230	µ		

Codifica delle informazioni

- Una codifica è una convenzione, come visto in precedenza, quindi la sequenza di bit **01001100 può rappresentare il carattere L (L maiuscola) oppure, se invece intendiamo un valore intero senza segno, rappresenta il numero decimale 76 .**
- Ad esempio , **ogni immagine è composta da pixel. Se usassi solo 1 bit per ogni pixel, potrei avere solo immagini in bianco e nero (1 pixel nero, 0 pixel bianchi).**

Codifica delle informazioni

Se utilizzo più bit per rappresentare ciascun pixel, posso ottenere gamme di grigi o colori. E da lì, suoni, video...



Il pixel rappresenta il più piccolo elemento autonomo dell'immagine. Ogni pixel è quindi caratterizzato da posizione propria

Il numero totale di pixel in un'immagine digitale è chiamato **risoluzione** . Ad esempio, se ho una griglia 10 x 10, l'immagine sarà composta da 100 pixel.

dpi = **numero di punti per pollice,** ad esempio in un monitor tipico avrò 72 pixel per pollice

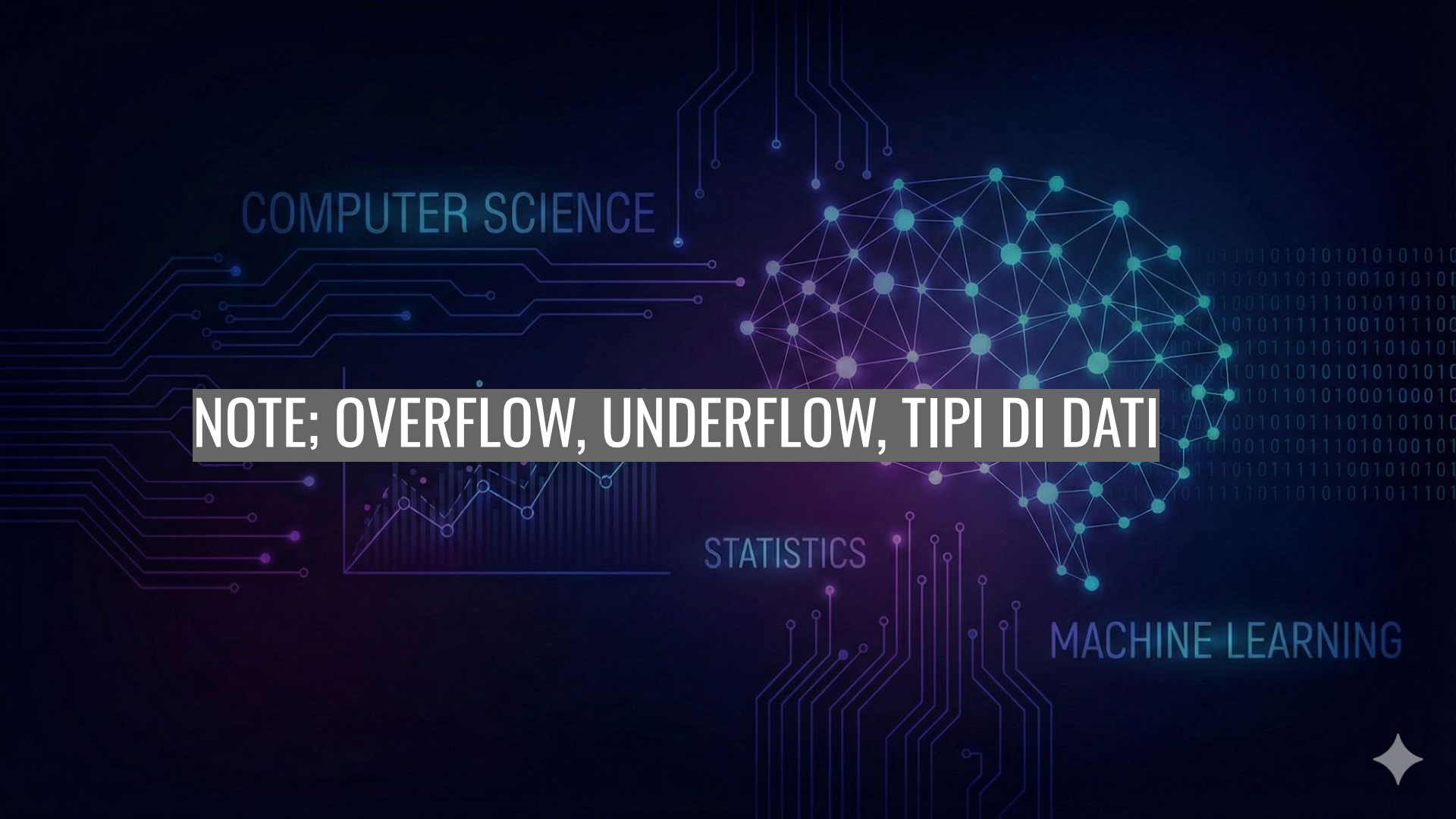
Profondità: nel caso di un'immagine in scala di grigi, si possono utilizzare 8 bit per ogni pixel, avendo quindi a disposizione 2^8 = **256 sfumature di grigio.**

Codifica delle informazioni

- immagini a colori

Nel caso delle immagini a colori, **ogni pixel** è caratterizzato da **tre scale di colore** per i tre colori primari, **RGB (rosso, verde, blu).**

Ad esempio, **un'immagine a 8 bit** avrà 256 possibili sfumature di rosso, 256 possibili sfumature di verde e 256 possibili sfumature di blu per ogni colore. Pertanto, un totale di **1.677.7216** possibili sfumature di colore.
Nel caso di immagini a 12 bit (quelle professionali di alta qualità), avremo invece 2^{12} = **4.096 sfumature di per ogni colore primario** , per un totale di **4.096 x 4.096 x 4.096** = **68.719.476.736** colori possibili per ogni pixel.



NOTE; OVERFLOW, UNDERFLOW, TIPI DI DATI

La memoria dei computer si sta esaurendo

- **OVERFLOW** : Non ci sono abbastanza bit per rappresentare il risultato. **UNDERFLOW:** Numero troppo piccolo 3/2, non posso rappresentare come intero il risultato 1,5, posso rappresentare solo 1 o 2 se lavoro con interi.
- Ad esempio, se utilizzo 8 bit per rappresentare i numeri interi positivi, posso rappresentare solo i numeri da 0 a 255, quindi cosa succede se provo a sommare 1 a 255?

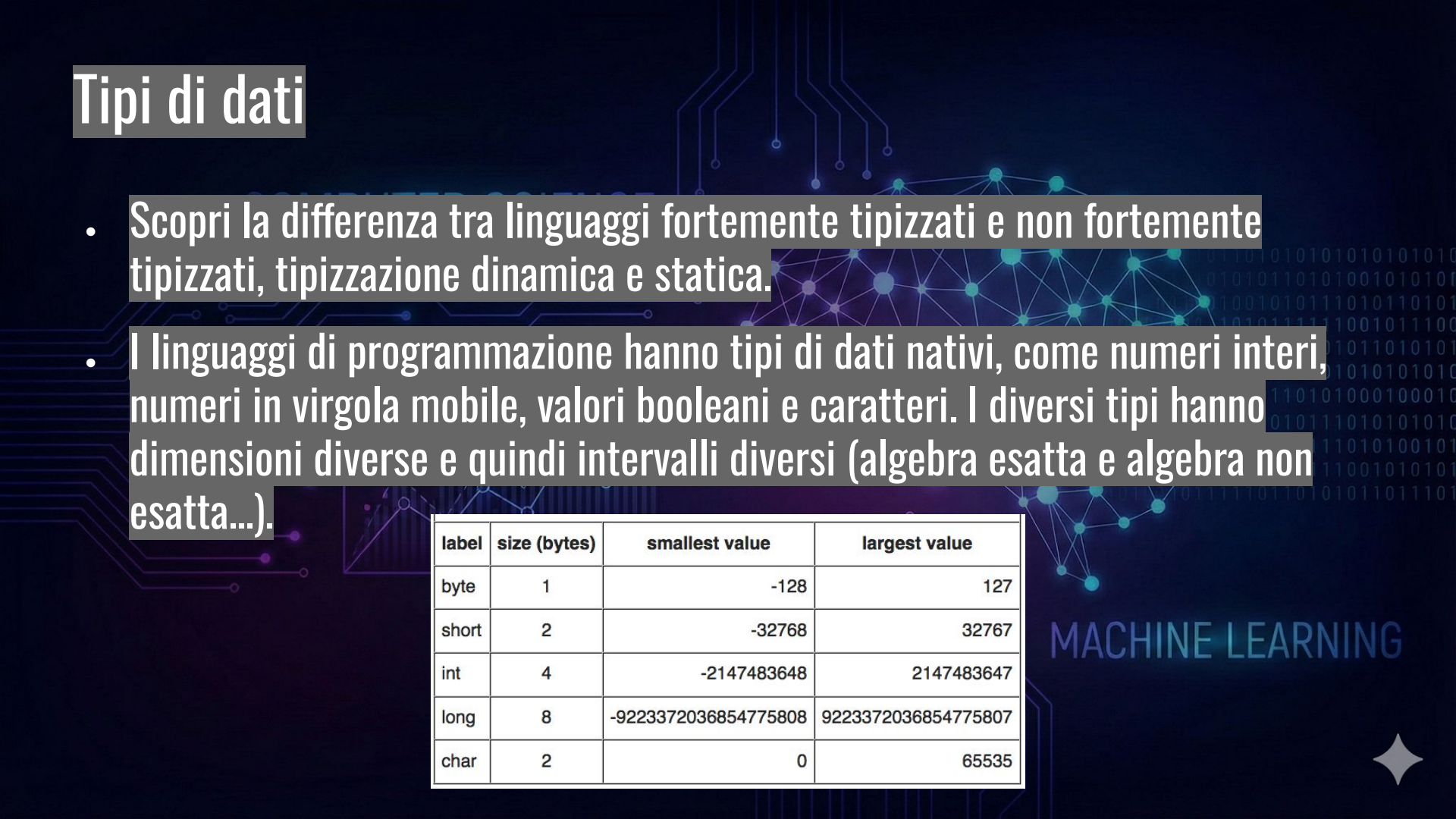
```
  11111111 +
  00000001 =
  ----------
 100000000
```

per rappresentare il risultato 8 bit non sono sufficienti

Tipi di dati

- Scopri la differenza tra linguaggi fortemente tipizzati e non fortemente tipizzati, tipizzazione dinamica e statica.
- I linguaggi di programmazione hanno tipi di dati nativi, come numeri interi, numeri in virgola mobile, valori booleani e caratteri. I diversi tipi hanno dimensioni diverse e quindi intervalli diversi (algebra esatta e algebra non esatta...).

label	size (bytes)	smallest value	largest value
byte	1	-128	127
short	2	-32768	32767
int	4	-2147483648	2147483647
long	8	-9223372036854775808	9223372036854775807
char	2	0	65535

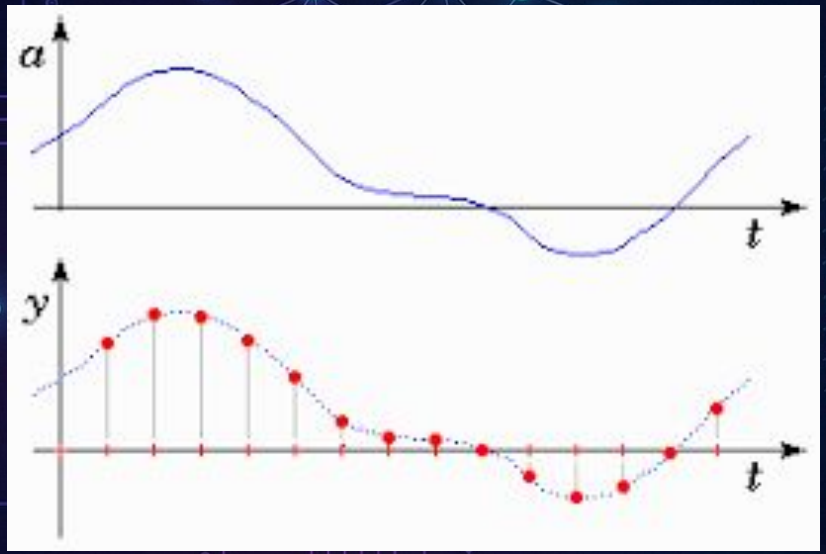


NOTE SULLA CONVERSIONE ANALOGICO-DIGITALE

Segnale analogico

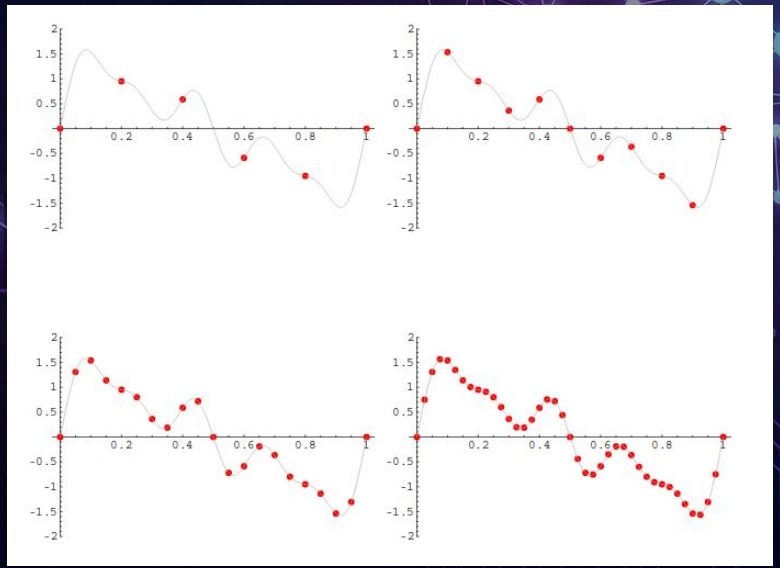
Conversione da analogico a digitale

Campionamento - discretizzazione temporale (teorema del campionamento di Nyquist-Shannon)
quantizzazione - discretizzazione dell'ampiezza
codifica - utilizzo di "parole" binarie per esprimere il valore del segnale



Segnale analogico

Ovviamente, la fedeltà migliora all'aumentare del numero di campioni per unità di tempo (frequenza di campionamento) e del numero di livelli di quantizzazione.



Audio digitale

- Poiché l'orecchio umano è in grado di percepire frequenze in un determinato intervallo (circa 20-20000 Hz), il teorema del campionamento ci dice che dovremmo campionare a 40 kHz.
- In genere, si utilizza un numero di livelli di quantizzazione molto superiore a 256, spesso a 16 bit.
- Ad esempio, nel caso di un **CD** abbiamo due canali (stereo) a (l'ingegneria pratica richiede una frequenza leggermente superiore a 40 kHz) **44,1 kHz e 16 bit (2^{16} = 65536)**
  - Quindi il **bit rate = 44100 campioni/s * 16 bit * 2 canali = 1411200 bit/s = 176400 byte/s**
  - Se vogliamo **1 minuto** di musica, abbiamo bisogno **di 60*176400 Byte/s = 10584000 Byte** , ovvero circa 10 MiB.

Considerazioni finali

- La compressione di immagini e suoni (e quindi anche di video) si basa essenzialmente sull'eliminazione delle informazioni alle quali l'orecchio e l'occhio umano non sono molto sensibili.
- Ma non solo...

NOZIONI DI BASE DI CRITTOGRAFIA

Nozioni di base sulla crittografia

- Nei computer, le informazioni vengono memorizzate come sequenze di bit.
- **Le tecniche crittografiche modificano queste sequenze (stringhe) per ottenere sequenze diverse che possono poi essere trasmesse e ritrasformate dal destinatario nella sequenza originale** . Le funzioni matematiche utilizzate nella trasformazione si avvalgono di una o più chiavi segrete.
  - **Crittografia simmetrica:** utilizzata fin dai tempi degli Egizi e degli antichi Romani
  - **Crittografia asimmetrica** : risale agli anni '70

Terminologia di base

Per comprendere il funzionamento della messaggistica sicura, dobbiamo definire i due stati del ciclo di vita di un messaggio.

Testo semplice

Il messaggio o i dati originali e intelligibili che vengono inseriti nell'algoritmo. Sono completamente leggibili da chiunque li intercetti.

Esempio: `"MEET ME AT NOON"`

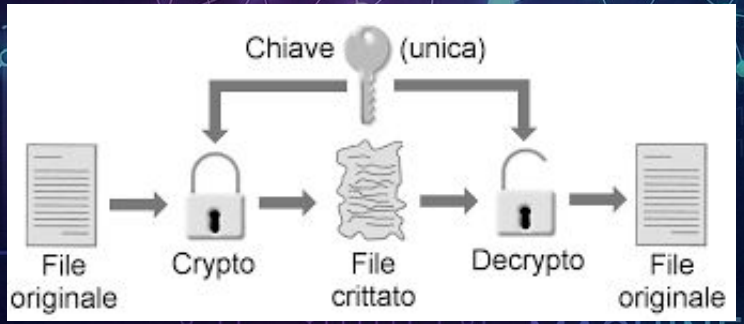
Testo cifrato

Il risultato, apparentemente casuale e disordinato, dell'algoritmo di crittografia. È completamente illeggibile senza la chiave matematica corretta.

Esempio: `"X#9!Qp2L@Z$8"`

Crittografia simmetrica

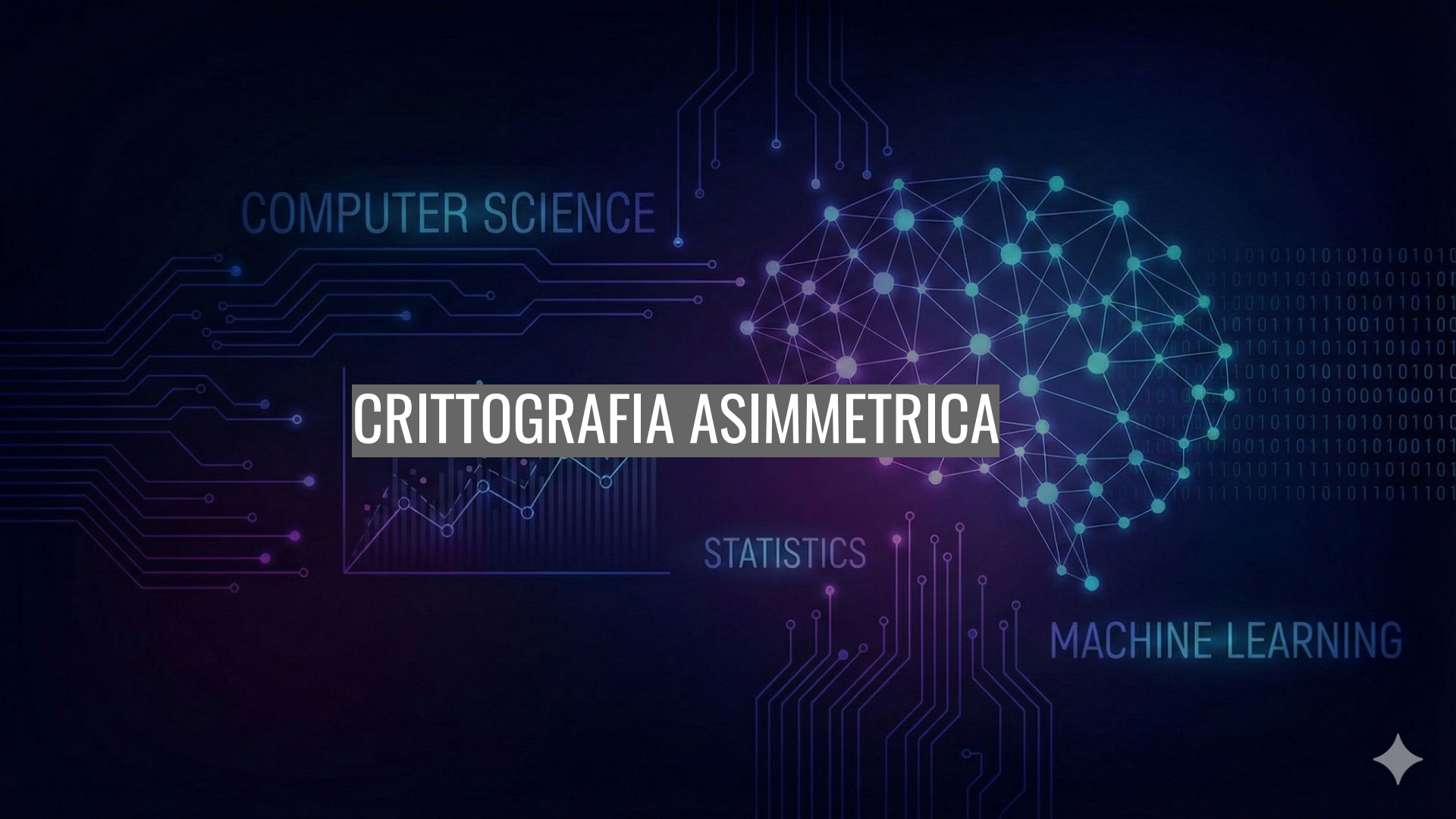
- **La chiave utilizzata per la crittografia e la decrittazione, e quindi per il mittente e il destinatario, è unica** . Ad esempio, nel cifrario di Cesare, ogni carattere viene sostituito con un altro carattere spostato di k posizioni (la chiave è il valore di k).
- cifrario monoalfabetico



a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
m	n	b	v	c	x	z	a	s	d	f	g	h	j	k	l	p	o	i	u	y	t	r	e	w	q

Crittografia simmetrica

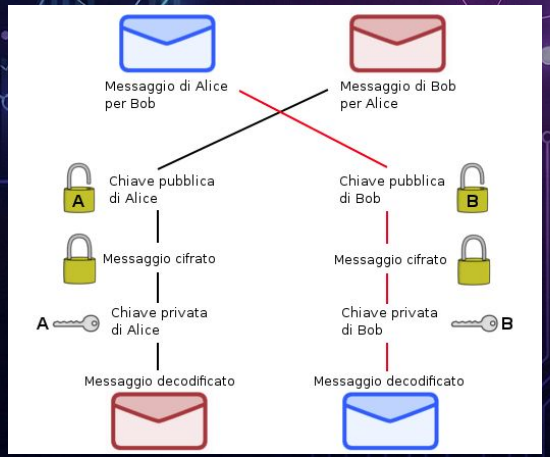
- **Devo trovare un modo sicuro per scambiare la** chiave segreta, che è univoca sia per il mittente che per il destinatario.
- **bruta** (ad esempio nel caso del cifrario di Cesare) Provo tutti i valori di k e che vedo quando ottengo frasi e parole "corrette".
- Esempi di algoritmi moderni: **Blowfish, Twofish, DES standard o Triple DES, AES standard** . Sono tutti basati su problemi matematici difficili da risolvere senza conoscere la chiave segreta.



CRITTOGRAFIA ASIMMETRICA

Crittografia asimmetrica

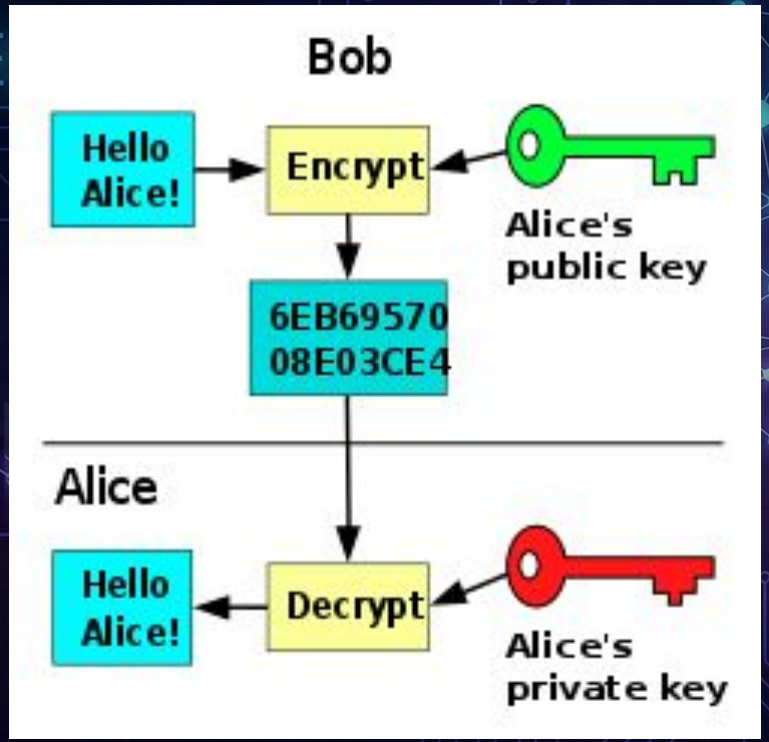
Le chiavi utilizzate per la crittografia e la decrittazione sono diverse . La chiave privata, che deve essere tenuta segreta, viene utilizzata per decrittografare e quindi per recuperare il messaggio originale; la chiave pubblica viene utilizzata per crittografare.



Crittografia asimmetrica

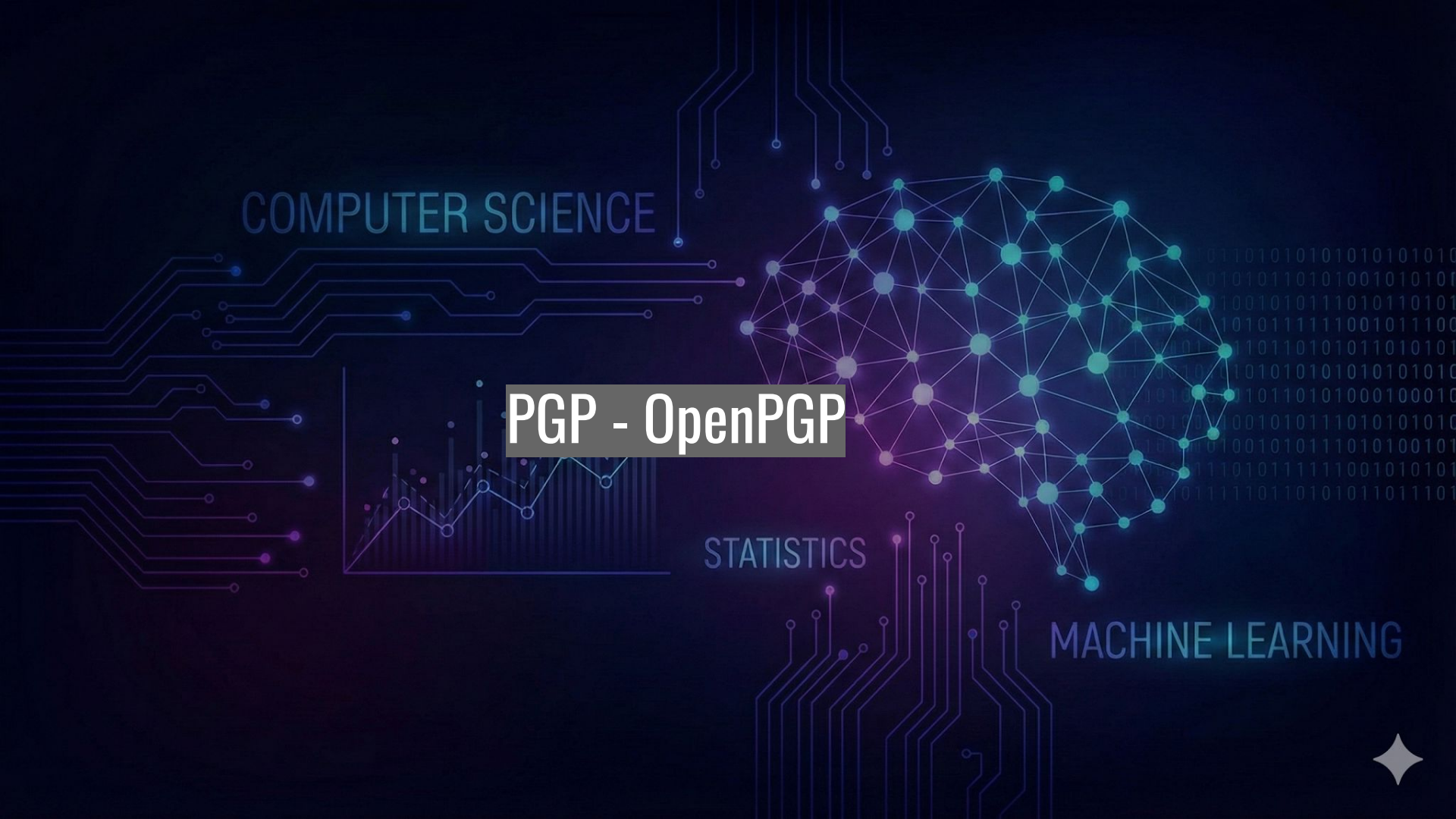
- **Quando l'utente genera la coppia di chiavi, deve custodire gelosamente la** chiave privata (segreta) **KS** e distribuire invece solo quella **pubblica KP . KS ad esempio in una smart card** e in tal caso la smart card stessa eseguirà la crittografia.
- Se l'utente Bob desidera inviare un messaggio privato all'utente Alice, Bob utilizzerà la chiave pubblica di Alice, KP, e invierà il **crittogramma risultante** . Solo Alice, che possiede la parte privata della chiave, KS, sarà in grado di recuperare il messaggio originale (in chiaro).
- La crittografia asimmetrica viene utilizzata anche nei processi di autenticazione.

Crittografia asimmetrica



Crittografia asimmetrica: RSA

- L'algoritmo più conosciuto e utilizzato è **RSA** (nomi degli inventori: Rivest, Shamir, Adleman).
- Anche per l'autenticazione o per garantire l'integrità di un documento (compresa la firma digitale).
- **Basato sui numeri primi, cioè quei numeri naturali divisibili solo per 1 o per se stessi (2, 3, 5, ..., 19249 - $2^{13018586} + 1$)**
- **In pratica, la sicurezza si basa sulla difficoltà di trovare i fattori primi di un numero grande (il modulo). KS e KP sono derivati matematicamente da questi fattori.**
- Interesse per i numeri primi e gli algoritmi di fattorizzazione (**algoritmo di Shor per i computer quantistici**)

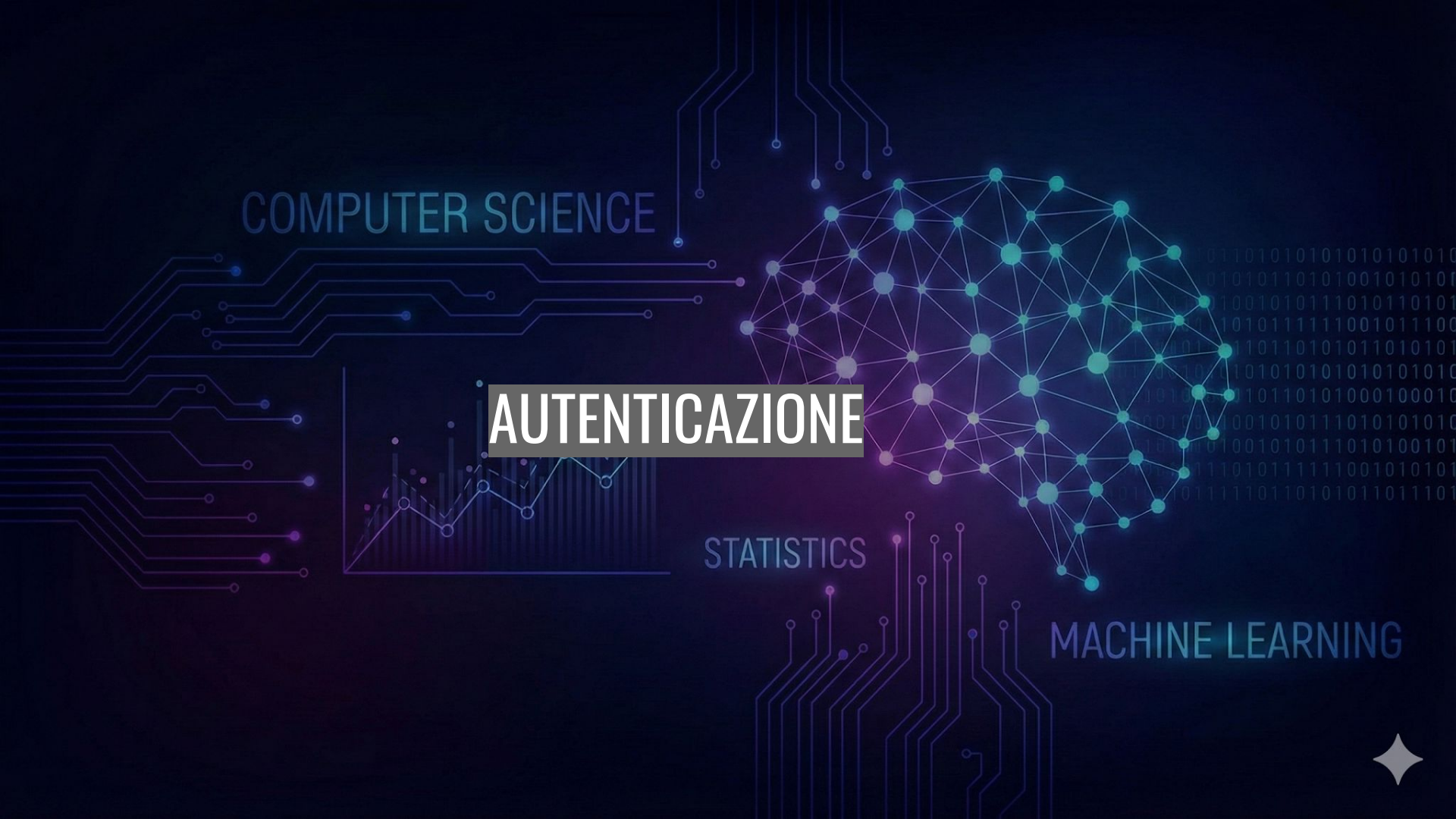

COMPUTER SCIENCE
PGP - OpenPGP
STATISTICS
MACHINE LEARNING

OpenPGP

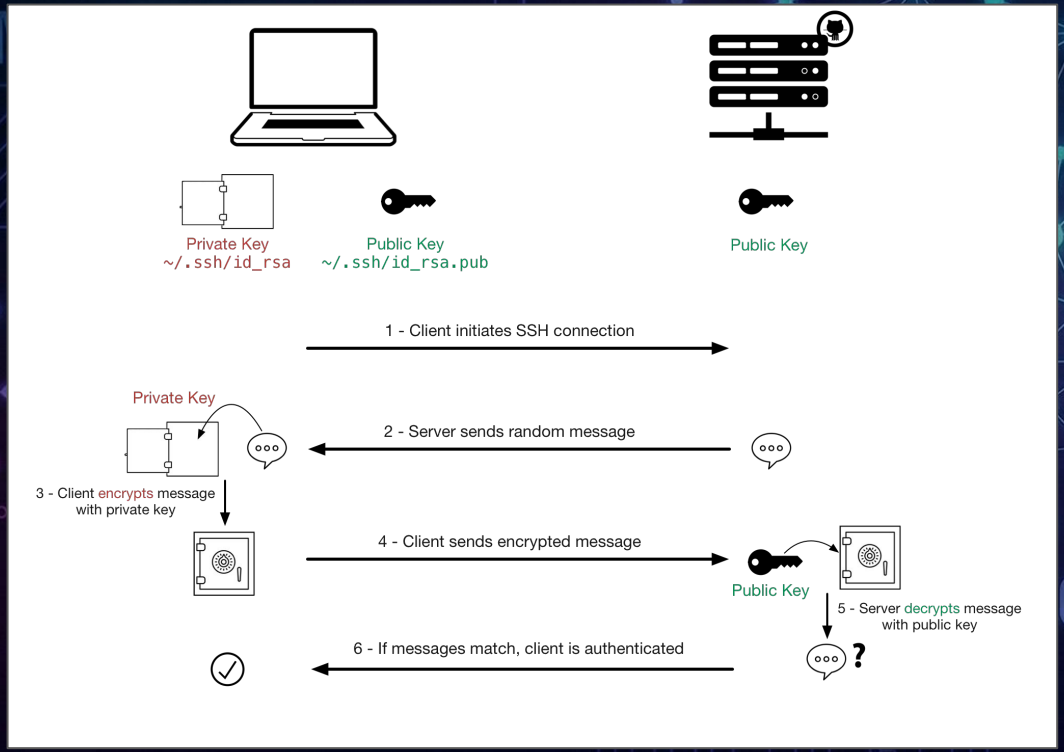
- RSA è un algoritmo (in realtà, due algoritmi: uno per la crittografia asimmetrica e uno per le firme digitali, con diverse varianti). **PGP** è un software, **ora un protocollo standard, generalmente noto come OpenPGP** .
- **OpenPGP definisce formati per elementi di dati che supportano la messaggistica sicura** , con crittografia e firme, e varie operazioni correlate come la distribuzione delle chiavi.
- Come protocollo, **OpenPGP si basa su una vasta gamma di algoritmi crittografici.** Tra gli algoritmi che OpenPGP può utilizzare c'è **RSA** .

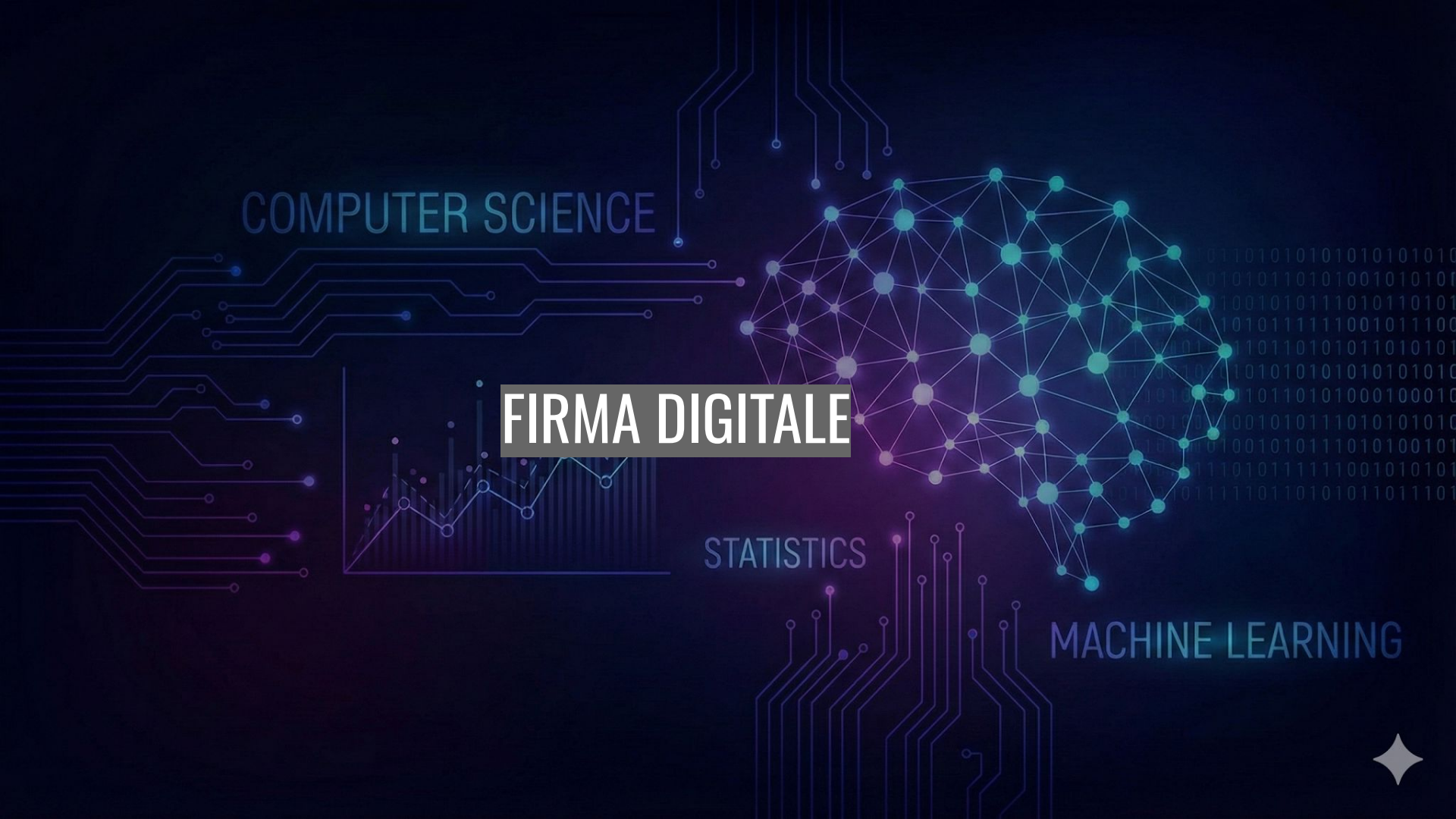
OpenPGP

- **Philip R. Zimmermann** è il creatore di **Pretty Good Privacy** , un software di crittografia per le email. Originariamente concepito come strumento per la tutela dei diritti umani, **PGP** è stato **rilasciato gratuitamente su Internet nel 1991** .
    - Questo rese Zimmermann bersaglio di un'indagine penale durata tre anni, poiché il governo riteneva che le restrizioni statunitensi all'esportazione di software crittografico fossero state violate con la diffusione mondiale del PGP.
- **GNU Privacy Guard** (GnuPG o GPG) è un software libero progettato per sostituire la suite crittografica PGP.


COMPUTER SCIENCE
AUTENTICAZIONE
STATISTICS
MACHINE LEARNING

Esempio di autenticazione SSH




COMPUTER SCIENCE
FIRMA DIGITALE
STATISTICS
MACHINE LEARNING

Firma digitale

- Viene apposto sui documenti digitali per garantire
  - **Autenticità** : quindi garanzia dell'origine del messaggio
  - **Integrità** : il messaggio non è stato modificato in alcun modo
  - **Non ripudio** : la fonte del messaggio non può negare di averlo firmato.
- Solo il mittente può aggiungere quella particolare firma
- Chiunque può verificare chi ha firmato il messaggio (documento digitale, testo, audio, immagine, video).
- **Ingredienti base del sistema di crittografia asimmetrica e della funzione hash**

Firma digitale: metodi di firma

- **CADES** : Il file ha l' **estensione pdf.p7m** e può essere letto con software di firma digitale (File Protector, DiKe, ecc.).
- **PADES** : **estensione PDF** : il file è firmato con un software di firma e può essere letto con Acrobat Reader.
- **XADES:** estensione xml.p7m, il file xml viene letto automaticamente dal software che lo riceve

Che cos'è una funzione hash?

Una funzione hash è un algoritmo matematico fondamentale che prende un input (spesso chiamato "messaggio") di qualsiasi lunghezza e lo trasforma in una stringa di caratteri di lunghezza fissa.

Questo output è noto come **valore hash** , **digest** o semplicemente **hash** .

- **Input variabile:** è possibile eseguire l'hashing di una singola parola, di un'immagine 4k o dell'intera enciclopedia.
- **Output fisso:** l'hash risultante avrà sempre la stessa lunghezza, indipendentemente dalla dimensione dell'input.

Proprietà essenziali dell'hash

Affinché una funzione hash possa essere considerata crittograficamente sicura, deve rispettare rigorosamente tre regole.

Deterministico

Inserendo nello stesso identico messaggio di input l'algoritmo, si otterrà sempre, senza eccezioni, lo stesso identico hash di output. Non c'è alcuna casualità.

Resistenza pre-immagine

È una strada a senso unico. Deve essere computazionalmente impossibile decodificare o ricostruire il messaggio di input originale partendo solo dall'output dell'hash.

Resistente alle collisioni

Deve essere astronomicamente improbabile trovare due messaggi di input completamente diversi che producano esattamente lo stesso output hash.

Hashing contro crittografia

Questi termini vengono spesso confusi, ma servono a scopi crittografici completamente diversi.

Crittografia

Processo bidirezionale (reversibile)

La crittografia è progettata per nascondere le informazioni durante la trasmissione. Utilizza *chiavi matematiche* per codificare i dati e richiede le stesse chiavi per decifrarli e leggerli in seguito.

Scopo: Riservatezza e privacy.

Hashing

Processo unidirezionale (irreversibile)

L'hashing non utilizza chiavi. Una volta che i dati sono stati sottoposti ad hashing, non possono essere de-hashing. Viene utilizzato per generare un'impronta digitale univoca dei dati al fine di dimostrare che non sono stati modificati.

Scopo: Verifica e integrità.

L'effetto valanga

1 bit

CAMBIAMENTO MICROSCOPICO

Randomizzazione totale

Una caratteristica cruciale delle funzioni hash potenti è l' **effetto valanga** .

Se si apporta una modifica microscopica al file di input, anche solo invertendo un singolo bit binario, modificando un pixel in un'immagine o convertendo una lettera maiuscola in minuscola, si innesca una reazione a catena catastrofica all'interno dell'algoritmo.

L'hash risultante sarà completamente irriconoscibile rispetto all'hash precedente. Ciò impedisce agli aggressori di indovinare l'input analizzando gli schemi presenti nell'output.

Esempio di algoritmo: SHA-256

Algoritmo di hash sicuro (256 bit)

Sviluppato dalla NSA, SHA-256 è uno degli algoritmi di hashing più sicuri e diffusi al mondo. Il risultato è sempre una stringa esadecimale di 64 caratteri.

```
Input: "Hello"
185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969
```

```
Input: "hello" (h minuscola)
2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824
```

Notate come la modifica di una sola lettera possa stravolgere completamente il risultato!

Applicazioni nel mondo reale

Archiviazione delle password

I database non memorizzano mai la password effettiva. Memorizzano solo l' *hash* della password. Se un hacker viola il database, ruba solo hash inutili e irreversibili.

Firme digitali

Quando si scarica un software, le aziende forniscono un "checksum" (un hash). Si esegue l'hashing del file scaricato sul proprio computer; se corrisponde, il file è sicuro e non danneggiato.

Blockchain

Le criptovalute collegano blocchi di dati inserendo l'hash del blocco precedente all'interno di quello nuovo. Manomettere i dati precedenti distrugge la catena matematica.

Firma digitale: HASH

- Algoritmo **hash : MD5, SHA-1, RIPEMD, SHA-256:**
  - Una funzione che, dato un flusso di bit di dimensioni variabili, restituisce una stringa di lettere o numeri di dimensioni fisse (una sorta di marcatura monouso).
  - La stringa è un identificatore univoco; la modifica di un solo bit del flusso sorgente produce un HASH diverso.
  - Non è invertibile, quindi non è possibile determinare il flusso originale dalla stringa restituita.

Firma digitale: HASH

- Algoritmo **hash : MD5, SHA-1, RIPEMD, SHA-256:**
  - Una funzione che, dato un flusso di bit di dimensioni variabili, na

```
└─ $ ▶ date
dom 22 mar 2026, 16:28:42, CET
(base) |redo@rbuchner --> /home/redo/Downloads
└─ $ ▶ date  | sha256sum
a57ea038ba20eb66ba187ba674c3bc095dda473df43438c3c321409b196b0462  -
(base) |redo@rbuchner --> /home/redo/Downloads
└─ $ ▶ date  | sha256sum
5a6989fb0c3d9dc115e2791f2a948c330b1ae9ebdefb3c73c4fd32e143718b7f  -
(base) |redo@rbuchner --> /home/redo/Downloads
```

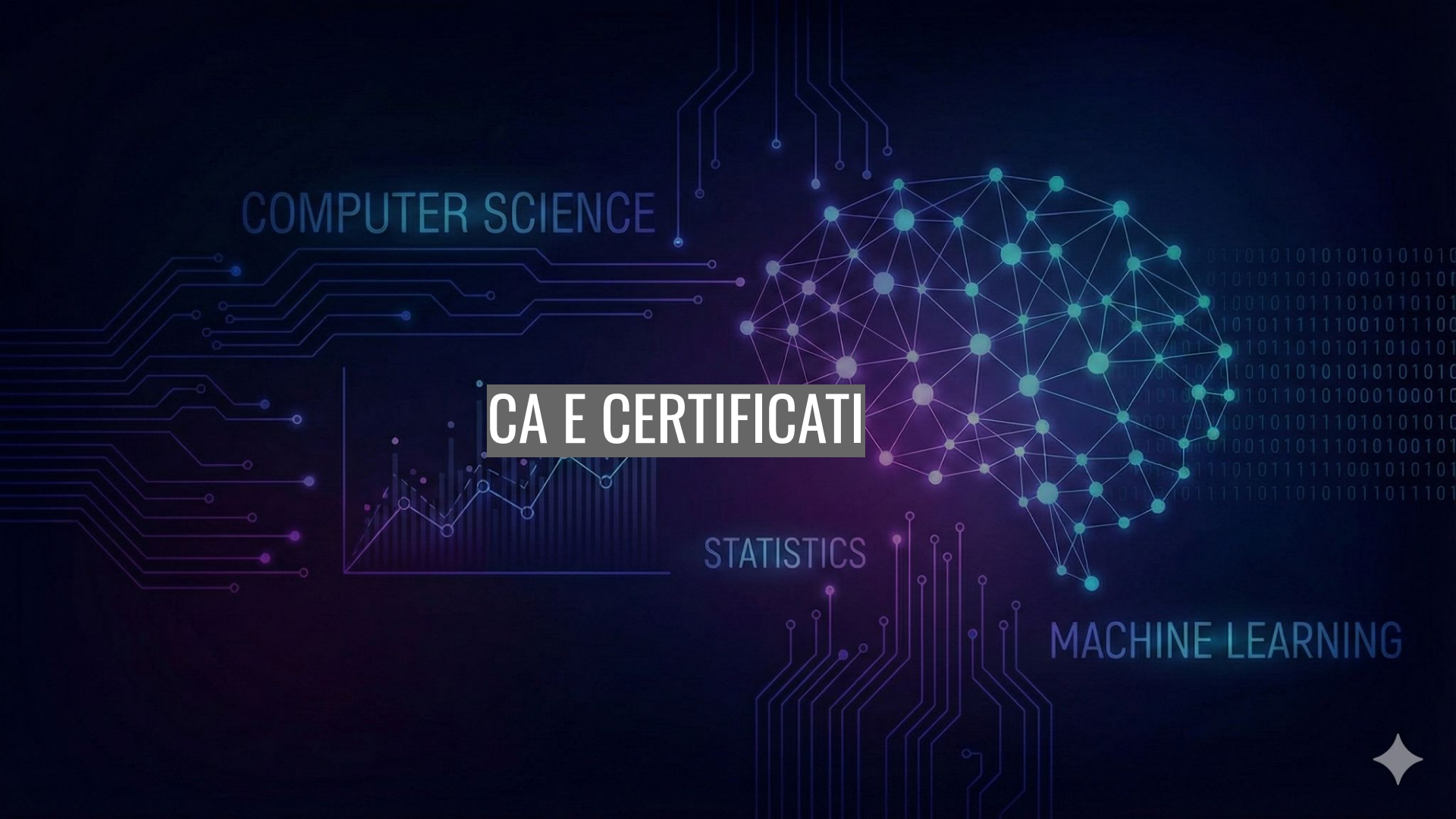
del

ale

dalla stringa restituita.

Firma digitale: come funziona

- Alice firmerà un determinato oggetto O (un documento, ad esempio):
  - **Calcola l'HASH di O** (detto anche **digest**)
  - **Crittografa l'HASH ottenuto con la tua chiave privata**
  - **Aggiungi l'HASH crittografato (la firma) insieme alla sua chiave pubblica** all'oggetto O, che chiameremo F.
- Bob verificherà la firma di Alice:
  - **Calcola l'HASH di O**
  - **Decifrare l'HASH crittografato** (ovvero la firma di Alice F trovata nel documento) **utilizzando la chiave pubblica di Alice .**
  - Ora **verifica che i valori siano gli stessi**


COMPUTER SCIENCE
CA E CERTIFICATI
STATISTICS
MACHINE LEARNING

CA e Certificati

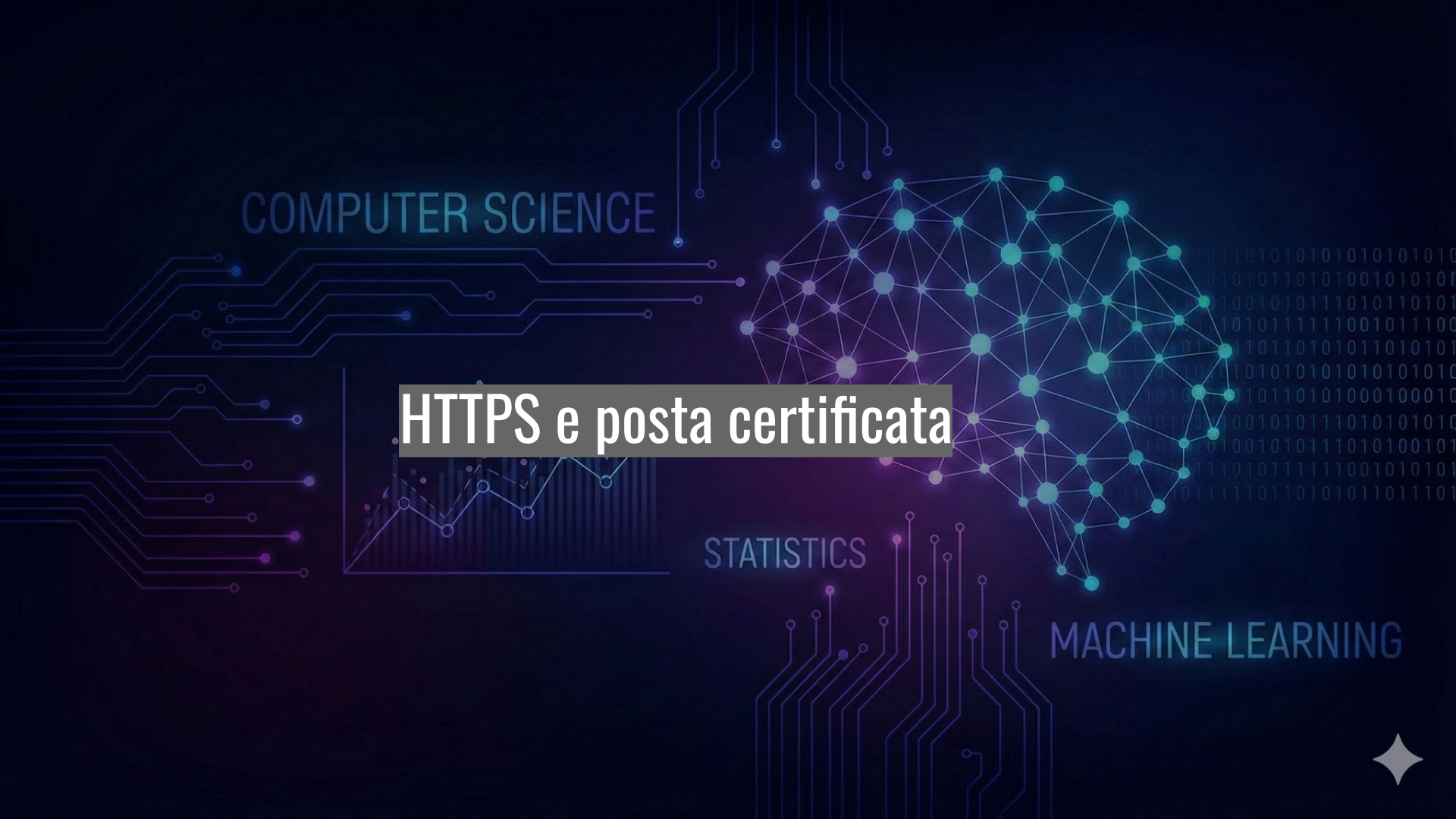
- Come posso essere sicuro che la firma utilizzata appartenga effettivamente al firmatario? In altre parole, come posso essere sicuro dell'associazione tra utente e chiave pubblica?
- **I certificati digitali servono a questo scopo. Contengono diverse informazioni, come la chiave pubblica e i dati dell'utente.**
- Proprio come i certificati cartacei ci permettono di avere informazioni sull'utente
- **CA, Autorità di Certificazione** , garantisce l'associazione tra la **Chiave Pubblica** e l'identità del proprietario.

CA e Certificati

- Certificati digitali: Consistono in (X.509):
  - **Chiave pubblica** del proprietario
  - H **rappresenta l'identità** (nome, cognome, data di nascita, ecc.).
  - **Data di scadenza** della chiave pubblica
  - **Nome dell'autorità di certificazione che l'ha emesso**
  - **Firma digitale dell'autorità di certificazione che ha emesso il certificato.**
- **Se ci fidiamo dell'Autorità di Certificazione (CA), possiamo verificarne la firma e quindi l'identità del firmatario.**

CA e Certificati

- **CA, Autorità di Certificazione** , garantisce l'associazione tra la firma digitale e l'identità del proprietario.
- Il certificato digitale è firmato da un'entità chiamata CA, che ne certifica l'autenticità. Il processo di firma viene eseguito dalla CA allegando le proprie informazioni di contatto al certificato e crittografando l'intero documento con la propria chiave privata.
- Se una determinata CA che firma un certificato non è considerata attendibile a livello locale, il sistema verifica il certificato di tale CA (emesso da una CA di livello superiore). Questo processo si ripete lungo la catena fino a raggiungere una CA radice esplicitamente considerata attendibile dal sistema, convalidando così l'intera catena.



HTTPS e posta certificata

HTTPS

- Protocolli fondamentali, ad esempio, nell'e-commerce e nell'home banking.
- Quando mi connetto a un sito tramite https:
  - **Il server dichiara la propria identità inviando** il certificato della chiave pubblica **garantito da una CA**
  - **Il mio browser (client) verifica che il nome host/dominio corrisponda all'identità contenuta nel certificato.**
  - **Nel moderno protocollo HTTPS (TLS 1.3), client e server utilizzano un metodo matematico chiamato Diffie-Hellman per generare la stessa chiave in modo indipendente. La chiave non viene mai trasmessa attraverso la rete, quindi anche se la chiave privata del server viene rubata in seguito, le conversazioni precedenti rimangono al sicuro.**

Posta Certificata

A differenza della posta elettronica standard, in cui si inviano semplicemente dati a un server, nel sistema PEC sia il Provider del mittente che quello del destinatario devono essere certificati e accreditati da un ente governativo (in Italia, AgID). Essi agiscono come garanti della comunicazione.

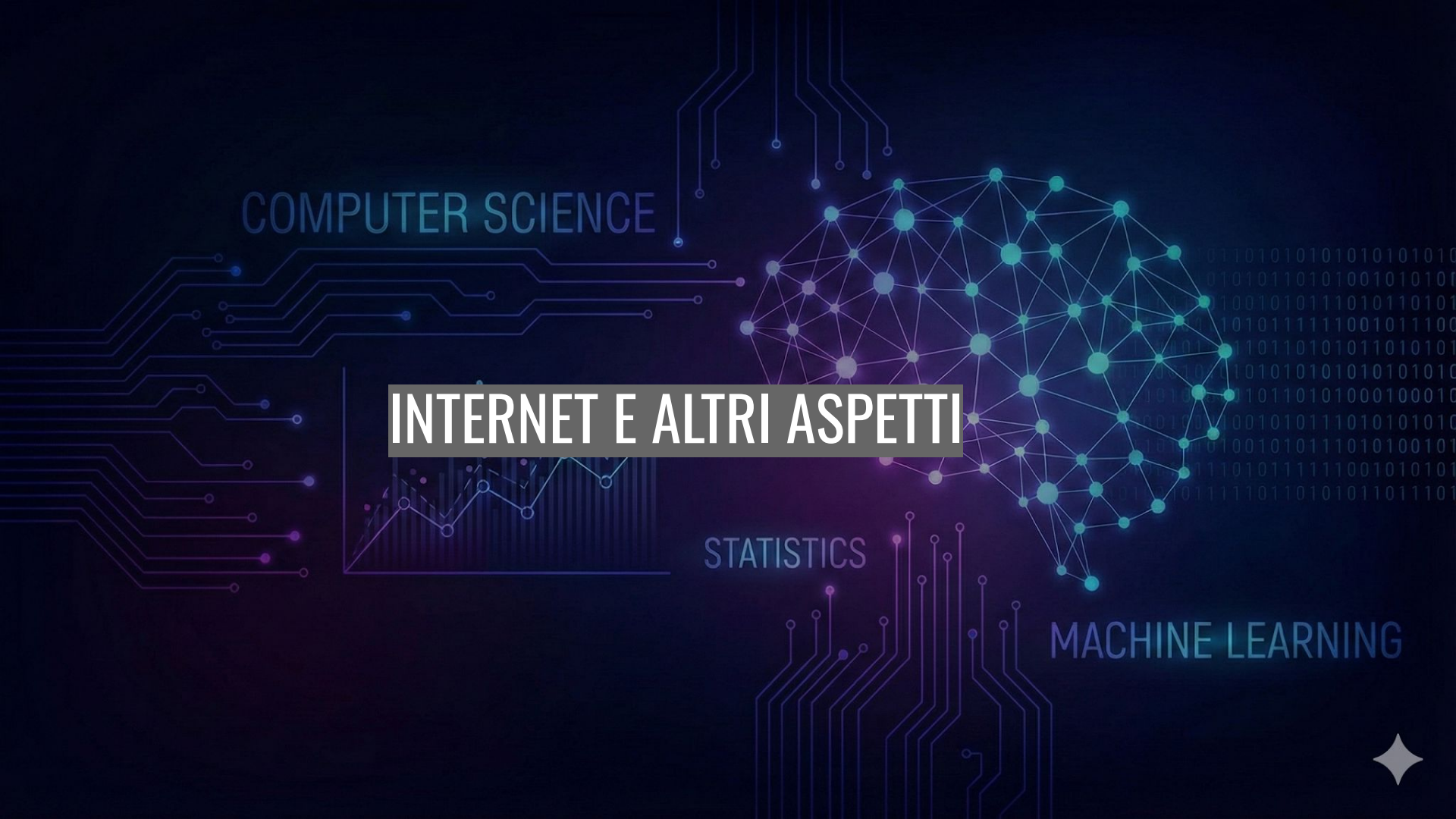
- Fase A: Presentazione e accettazione (Ricevuta di Accettazione)
  - Tu (mittente) invii l'e-mail al tuo fornitore PEC.
  - Il tuo provider controlla la posta elettronica per rilevare virus e verificarne la conformità.
  - **Il tuo Fornitore ti rilascia una Ricevuta di Accettazione.**
    - Cosa dimostra: Questa ricevuta è firmata digitalmente dal tuo fornitore. Prova legalmente quando (timestamp) hai inviato il messaggio e a chi l'hai inviato. Anche se il destinatario afferma di non averlo mai ricevuto, tu hai la prova dell'invio.

Posta Certificata

- Fase B: La Busta di Trasporto (Busta di Trasporto)
  - Il tuo provider non si limita a inoltrare la tua email. Avvolge il tuo messaggio originale all'interno di un nuovo pacchetto digitale chiamato Busta di Trasporto.
  - **Il fornitore appone la firma digitale su questa busta.**
    - **Cosa dimostra: Questo garantisce l'integrità. Il provider del destinatario** può verificare che il messaggio non sia stato alterato durante la trasmissione (nessun attacco Man-in-the-Middle).

Posta Certificata

- Fase C: Consegna (Ricevuta di Consegna)
  - Il tuo fornitore invia la busta al fornitore del destinatario.
  - Il fornitore del destinatario verifica la firma digitale sulla busta (controllando che provenga da un fornitore PEC valido).
  - Il provider del destinatario inserisce il messaggio nella casella di posta del destinatario.
  - Il fornitore del destinatario emette una Ricevuta di Consegna e la invia a te (mittente).
    - Cosa dimostra: Questa è la ricevuta più importante. Prova legalmente che il messaggio ha raggiunto l'indirizzo del destinatario. È l'equivalente legale della "ricevuta di ritorno" nella posta raccomandata tradizionale.
    - Nota: ciò dimostra l'avvenuta consegna nella casella di posta, non che l'utente l'abbia effettivamente letta. (Proprio come una raccomandata recapitata nella cassetta postale si considera legalmente notificata anche se non viene aperta).



INTERNET E ALTRI ASPETTI

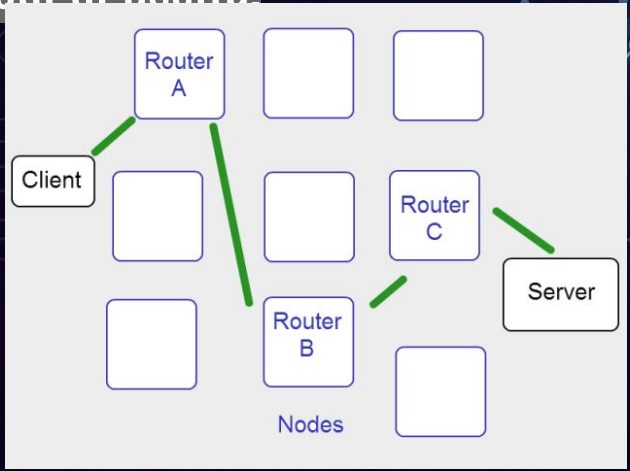
DEEP WEB, TOR

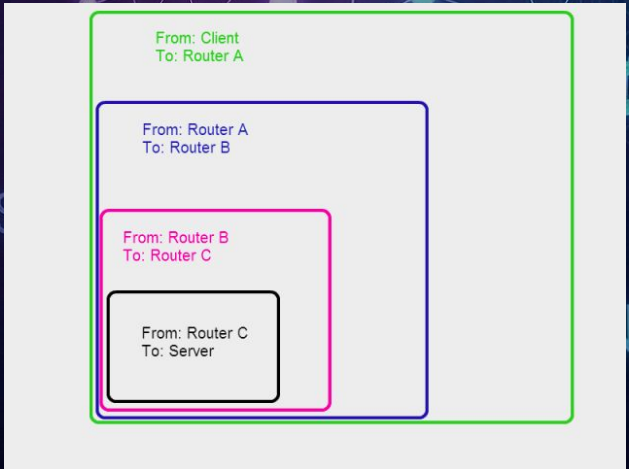
ONION ROUTING: A VOLUNTEER NETWORK FOR ANONYMITY

- I can reach “hidden” services, but also normal servers via exit nodes.

Deep Web, Tor

- Instradamento a cipolla. La rete Tor è composta da volontari che utilizzano i propri computer come nodi:
- Posso raggiungere i servizi “nascosti”, ma anche i server normali tramite nodi di uscita

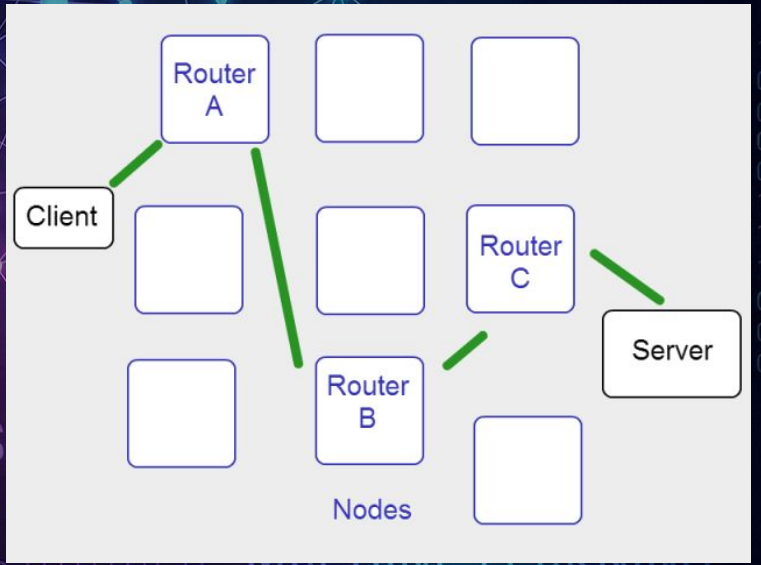




Deep Web, Tor

Server normali tramite nodi di uscita: se utilizzi Tor per navigare nel "Web normale", il tuo traffico esce dalla rete Tor attraverso un nodo di uscita.

1. Il client Tor costruisce un circuito casuale attraverso una serie di nodi volontari per connettersi al server di destinazione. Selezione del percorso: il client seleziona casualmente tre nodi da questo elenco per formare un "circuito":
   a. Nodo di ingresso (Guardia): Il primo hop. Vede il tuo indirizzo IP.
   b. Nodo intermedio: il secondo passaggio. Funge da buffer.
   c. Nodo di uscita: l'ultimo passaggio. Si connetterà al server di destinazione.

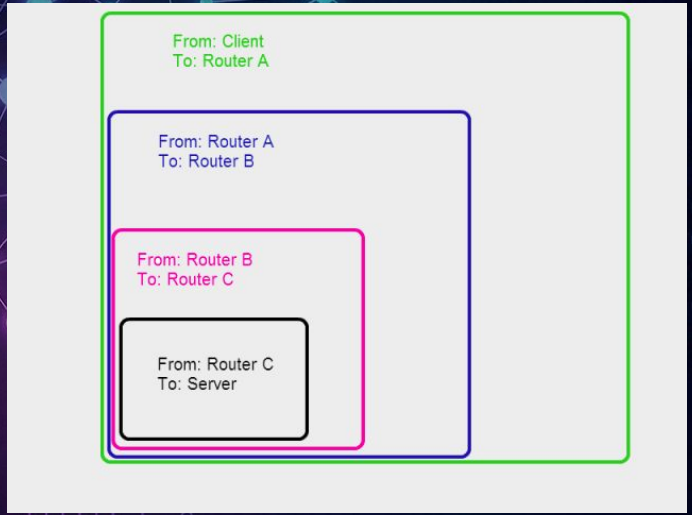


Deep Web, Tor

Crittografia a più livelli: il client crittografa i dati tre volte (come una cipolla):

1. Livello 1 (esterno): crittografato per il nodo di ingresso.
2. Livello 2 (intermedio): crittografato per il nodo intermedio.
3. Livello 3 (interno): crittografato per il nodo di uscita.

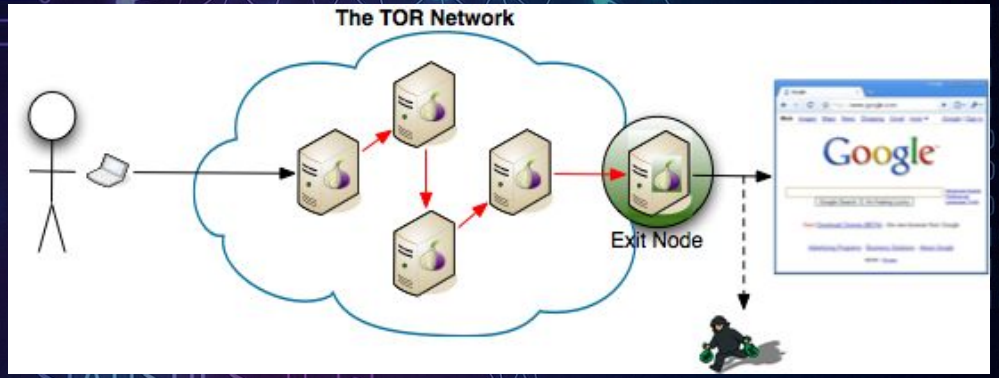
Il pacchetto di dati viaggia attraverso il circuito, venendo "scomposto" a ogni passaggio.



Deep Web, Tor, Bitcoin

Punto di vista del server: il sito web (Google) vede una richiesta proveniente dall'indirizzo IP del nodo di uscita. Non ha modo di conoscere il tuo indirizzo IP.

- Nodo di ingresso: vede il client (tu) e il nodo intermedio. → Vede il client!
- Nodo intermedio: vede il nodo di ingresso e il nodo di uscita. → Non vede né il client né il server.
- Nodo di uscita: vede il nodo intermedio e il server. → Vede il server!



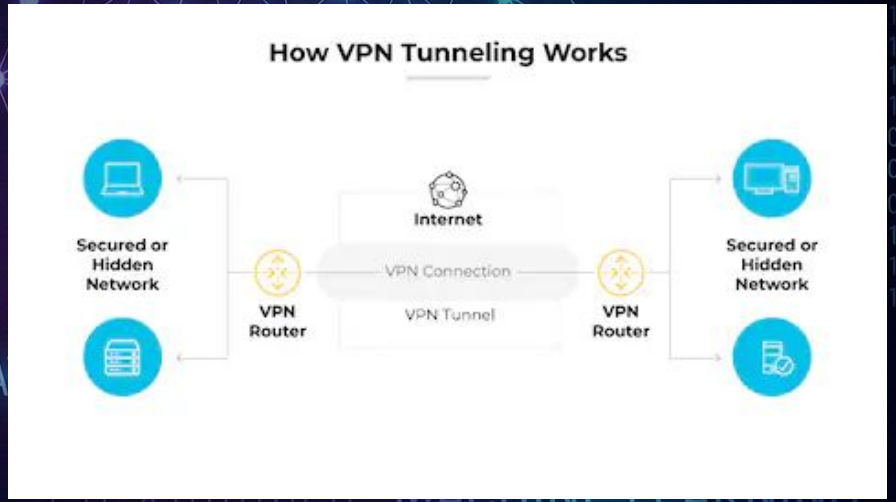
Deep Web, Tor, Bitcoin

Servizi nascosti di Tor (indirizzi che terminano con .onion).
Il server Onion vuole rimanere nascosto. Non pubblica il suo indirizzo IP.

- Punti di introduzione: Il server seleziona 3 relay casuali nella rete che fungeranno da "punti di introduzione".
- Crea circuiti crittografati con loro e dice: "Se qualcuno vuole parlare con me, inoltri il suo messaggio qui".

VPN: Rete privata virtuale

- Il tunnel sicuro: immaginate di guidare un'auto blindata privata su un'autostrada pubblica. La VPN crea una connessione crittografata sulla rete Internet pubblica, isolando i vostri dati dal traffico di terzi.
- Incapsulamento: i pacchetti di dati originali (lettera) vengono racchiusi in nuovi pacchetti VPN (busta). L'ISP vede solo la busta esterna, non la lettera al suo interno.
- Mascheramento IP: ti connetti a un server VPN e il server si connette al sito web. Il sito web vede l'indirizzo IP del server, non il tuo, nascondendo di fatto la tua posizione.



Bitcoin

- **Bitcoin** (cito Wikipedia) A differenza della maggior parte delle valute tradizionali, Bitcoin non si avvale di un'autorità centrale: utilizza un database distribuito tra i nodi della rete che tiene traccia delle transazioni, ma si serve della crittografia per gestire gli aspetti funzionali, come la generazione di nuova moneta e l'attribuzione della proprietà dei bitcoin.
- Basato sulla crittografia e sugli algoritmi di hashing (SHA-256), Bitcoin utilizza l'algoritmo di hash SHA-256 per generare numeri "casuali" verificabili in un modo che richiede un tempo di calcolo prevedibile. La generazione di un hash SHA-256 con un valore inferiore al target corrente risolve un blocco.

Blockchain

Il concetto chiave: un registro distribuito

Immaginate un quaderno (Registro/Libro mastro) che annota le transazioni.

- Sistema centralizzato (tradizionale): la banca conserva il registro contabile. Ci si fida della banca e si è certi che non modificherà i numeri.
- Blockchain: Ognuno possiede una copia del quaderno. Per scrivere una nuova pagina, la maggioranza della rete deve essere d'accordo.

Blockchain

Il blocco (il contenitore)

- Ogni blocco contiene tre elementi principali:
  - **Dati** : Le transazioni (ad esempio, "Alice invia 5 BTC a Bob").
  - **L'hash** : l'impronta digitale univoca del blocco (SHA-256).
  - **L'hash precedente** : questo è il collante magico. **Ogni blocco contiene l'hash del blocco precedente.**

Blockchain

La catena (l'anello)

- Poiché il Blocco 3 contiene l'hash del Blocco 2 e il Blocco 2 contiene l'hash del Blocco 1, i due blocchi sono matematicamente collegati.
  - Blocco 1 (Genesi): [Dati | Hash: A123 | Precedente: 0000]
  - Blocco 2: [Dati | Hash: B456 | Precedente: A123]
  - Blocco 3: [Dati | Hash: C789 | Precedente: B456]

Blockchain

Perché è sicuro? (Immutabilità) Se un hacker tenta di modificare una transazione nel Blocco 1:

- I dati nel Blocco 1 cambiano.
- Pertanto, l'hash del blocco 1 cambia (da A123 a X999).
- Il blocco 2 diventa immediatamente non valido perché riporta la dicitura "Precedente: A123", mentre il blocco precedente effettivo è ora "X999".
- Per risolvere questo problema, l'hacker dovrebbe ricalcolare gli hash per il Blocco 2, il Blocco 3 e ogni blocco successivo.