I principali cyber attacchi ai sistemi informatici della PA

Scuola Nazionale dell'Amministrazione
Vincenzo Calabrò

who am i

Formazione

- laureato in ingegneria informatica (università la sapienza di roma) e sicurezza informatica (università di milano)
- specializzato in advanced cybersecurity (stanford university)
- certificato in cybersecurity engineering and software assurance e digital forensics (carnegie mellon university)

Esperienza professionale

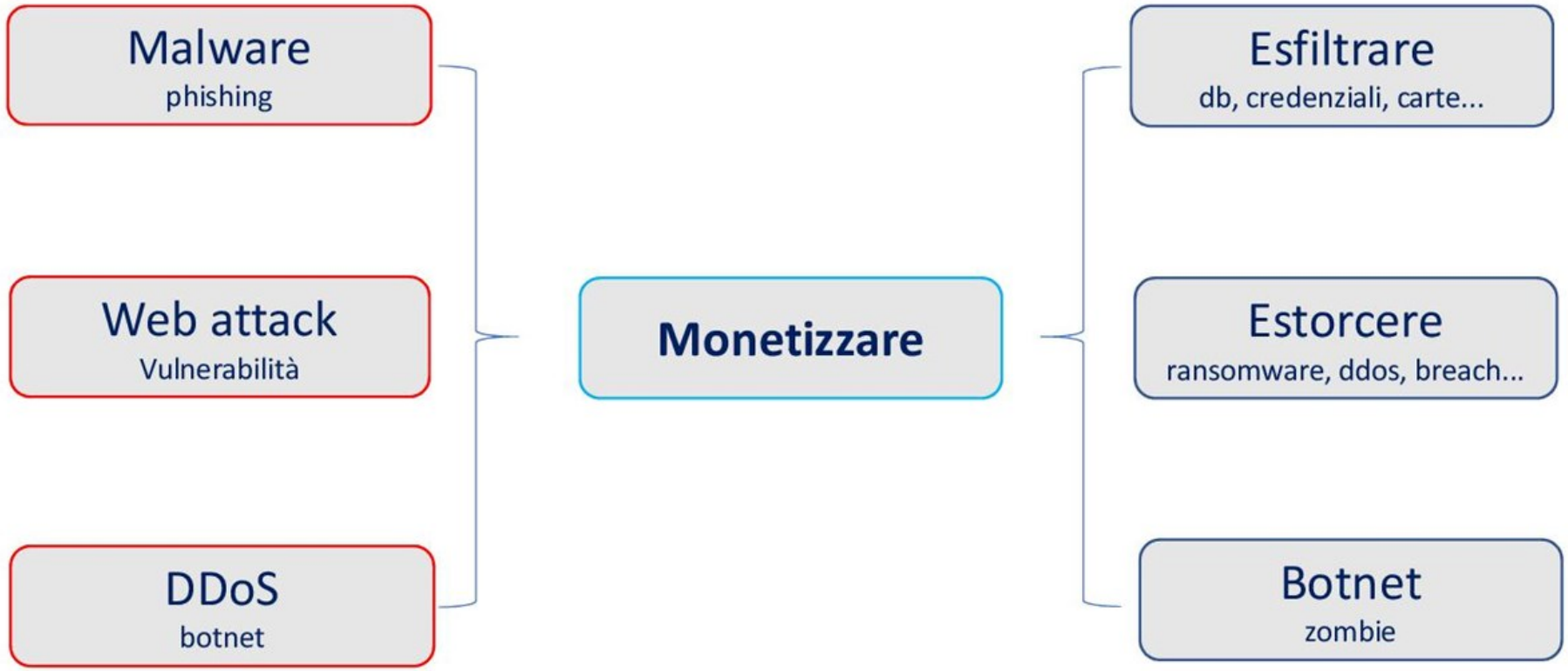
- funzionario alla sicurezza cis (ministero dell'interno)
- professore a contratto di sicurezza informatica
- consulente in sicurezza informatica e informatica forense
- relatore e autore sui temi della cybersecurity

LinkedIn vincenzocalabro.it

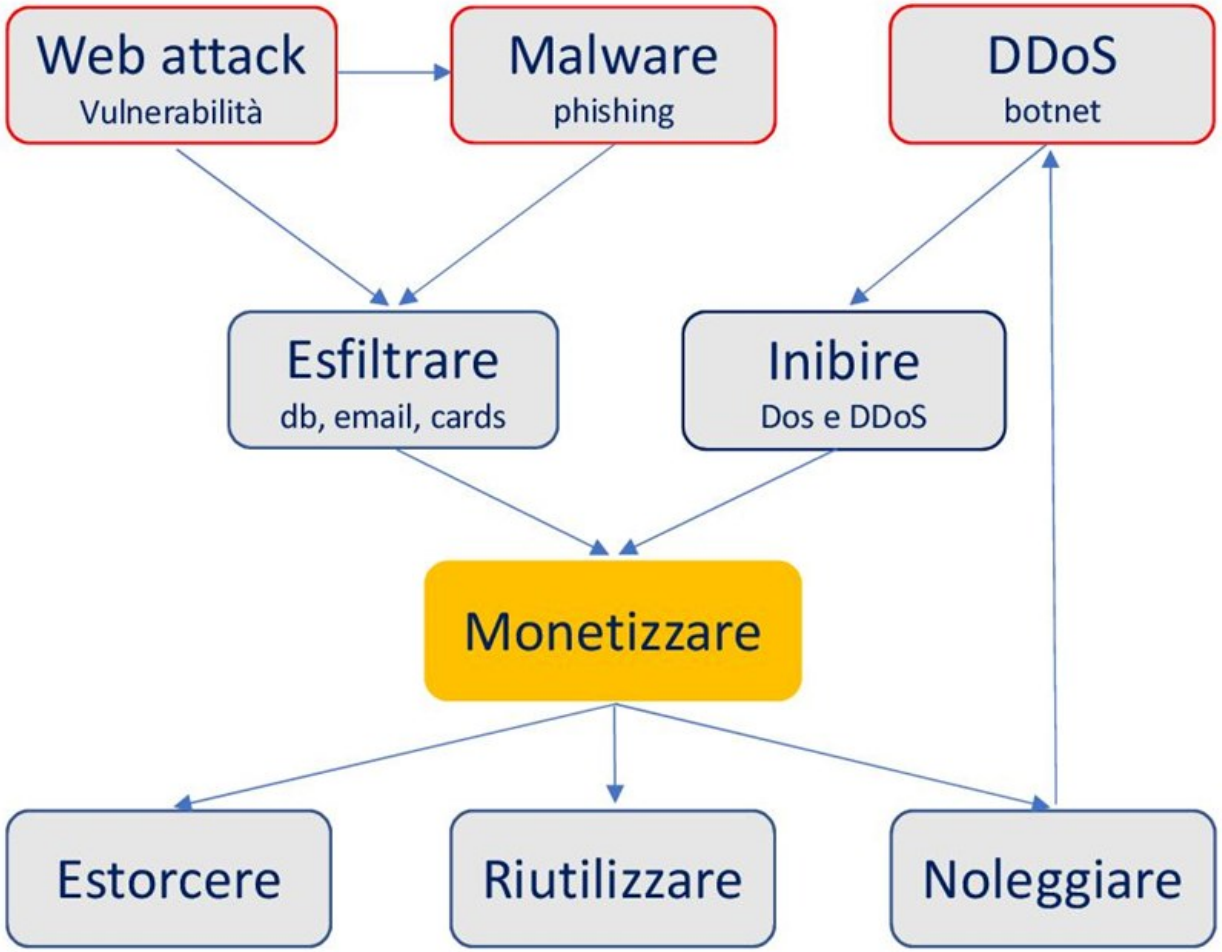
Gli attacchi informatici

Attività ostili nei confronti di una componente informatica, spesso compiute sfruttando le debolezze della componente umana.

Panorama delle minacce principali



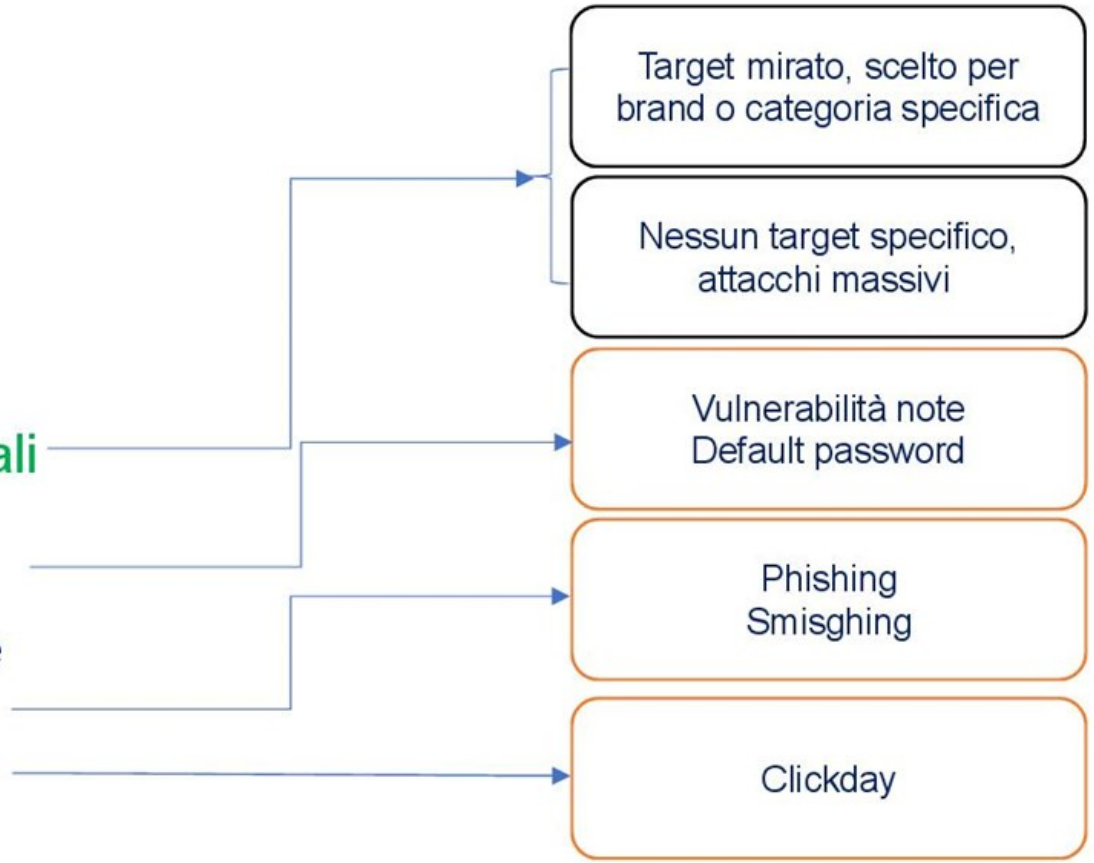
Flusso della minaccia in dettaglio



Gli attori

Le vittime, chi sono?

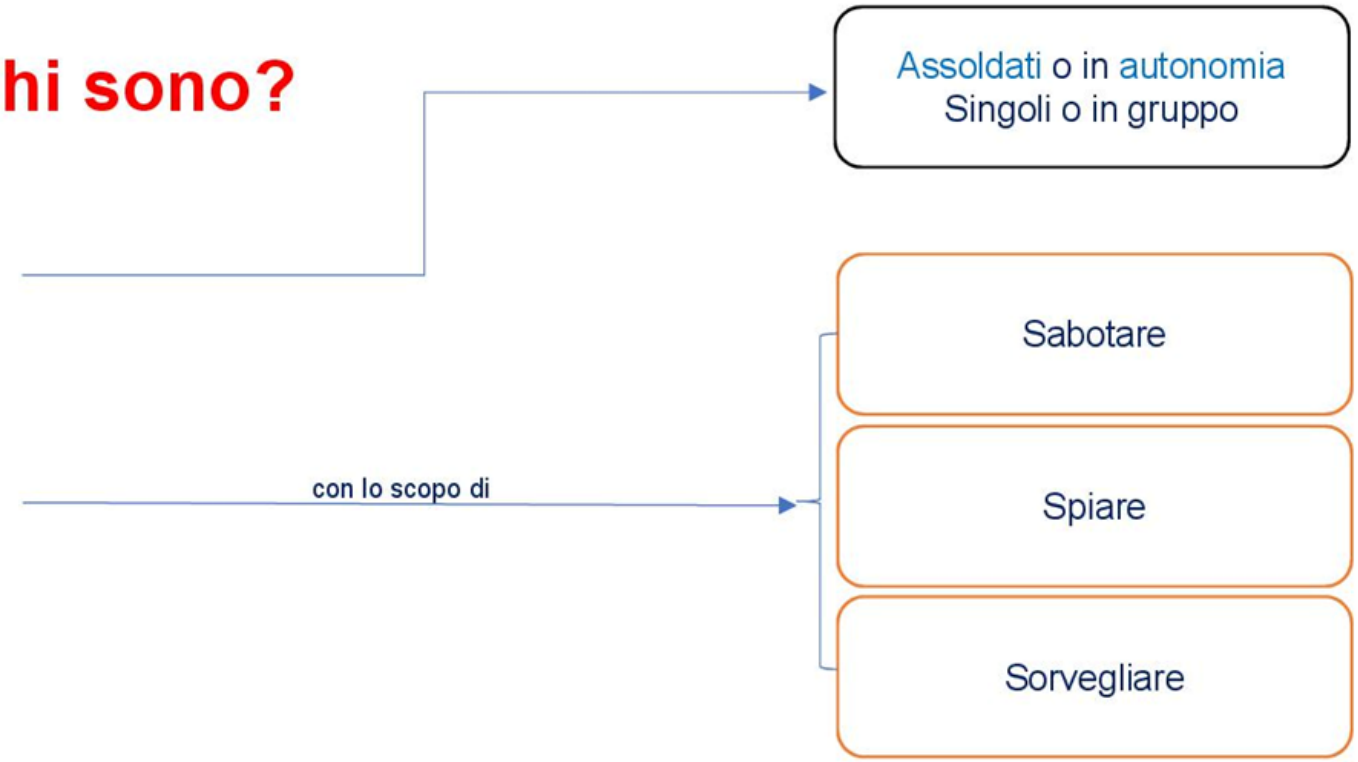
- Possono essere scelte o casuali
- Sistemi informatici esposti e vulnerabili
- Personale non adeguatamente preparato
- Eventi di interesse nazionale



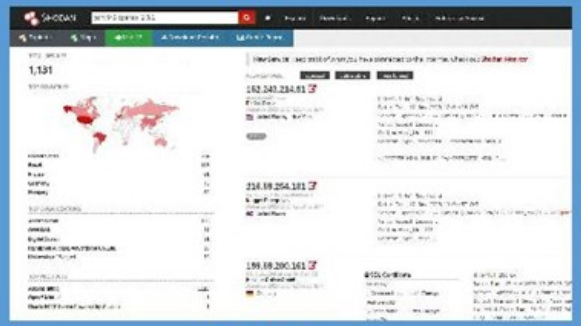
Gli attori

Gli attaccanti, chi sono?

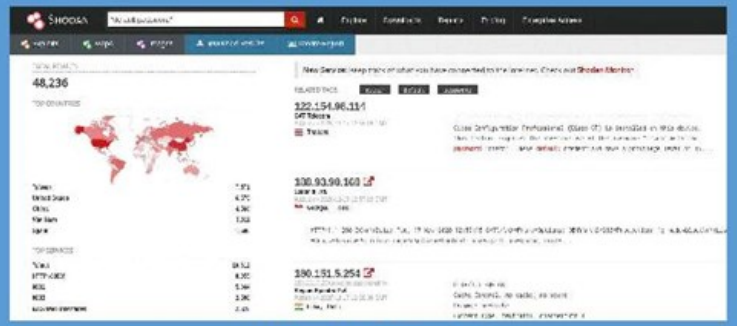
- Criminali di strada
- Hacktivisti
  anonymous
- Terroristi
  ISIS
- Paesi (ostili?)



Problemi per le vittime, risorse per gli attaccanti



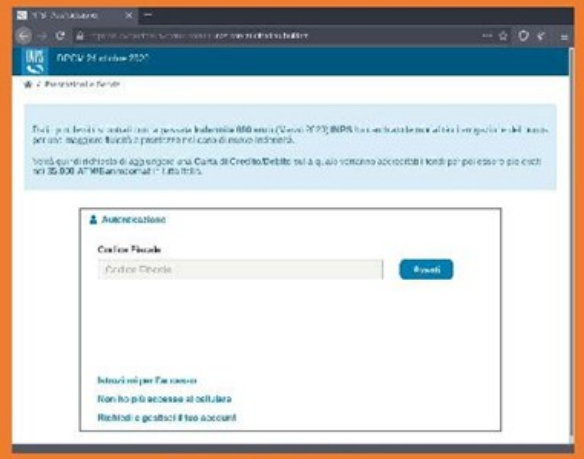
Heartbleed



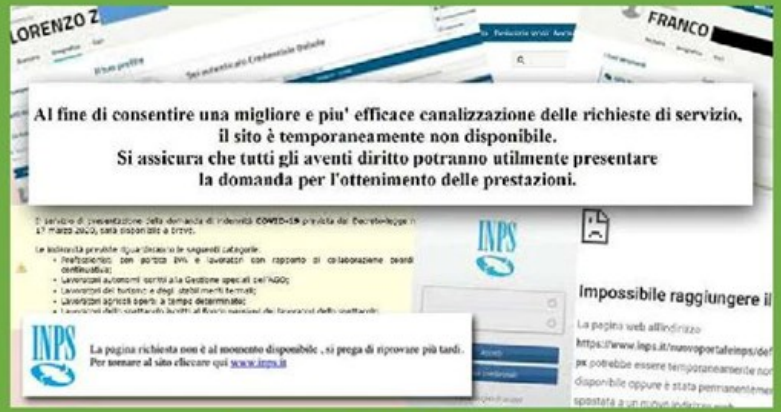
Default password

Deface

Malspam:Ursnif



Phishing: INPS



Clickday: INPS | Minambiente

Data Breach o Data Leak?

Data breach

Attacco mirato ad ottenere i dati privati di una organizzazione da parte di una entità non autorizzata.
Un data breach è solitamente dovuto ad una compromissione di un database o di credenziali di accesso ai dati della vittima.

Data leak

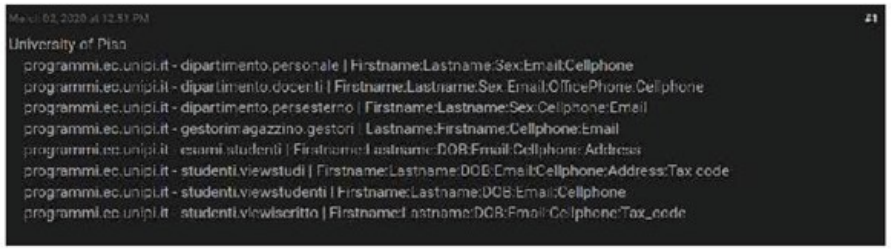
Trasmissione non autorizzata di dati da dentro una organizzazione verso l'esterno.
Le cause possono essere attribuite anche ad esposizione accidentale di informazioni dovute a *vulnerabilità* di tipo *Senstive Data Exposure* o ad errati processi aziendali di *conservazione dei dati*.

Data leak

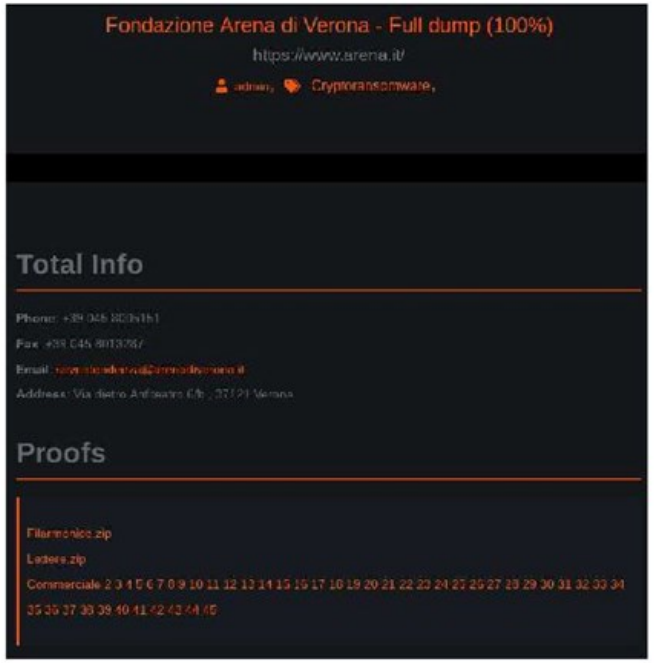
- Sono sempre esistiti ma oggi sembra essere diventata una moda
- È nato un mercato di nicchia in forte crescita
- I black market sono migrati dal dark al deep web
- Prezzi sempre più accessibili

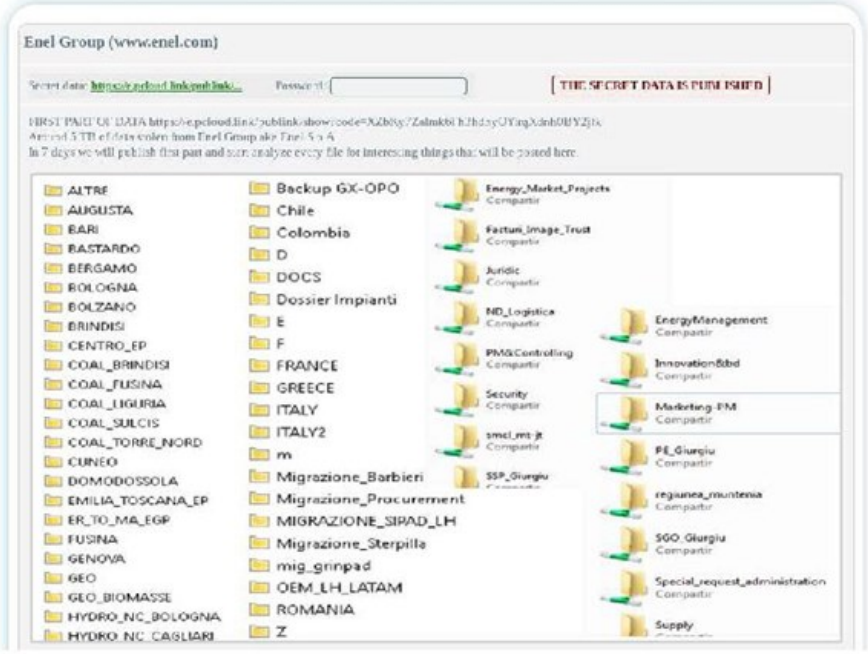
Quali dati in vendita? Ma soprattutto, sono sempre in vendita?

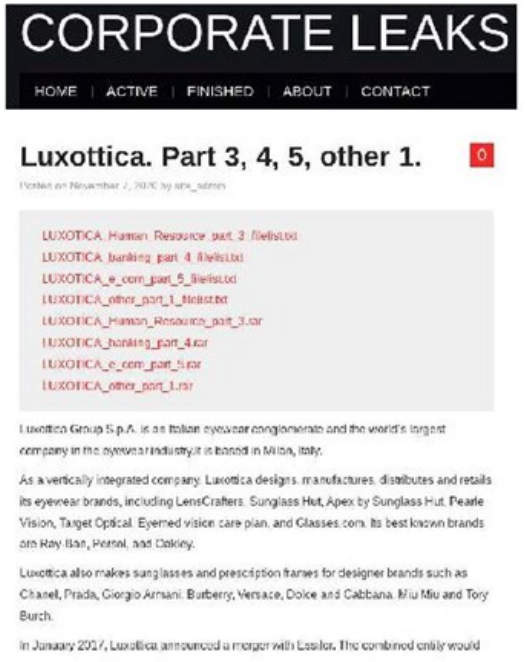
- Dati anagrafici
- Email
- Credenziali
- Carte di credito
- Metadati: *chi ha fatto cosa, quando e in quali circostanze*



Recenti leak con pubblica minaccia di estorsione



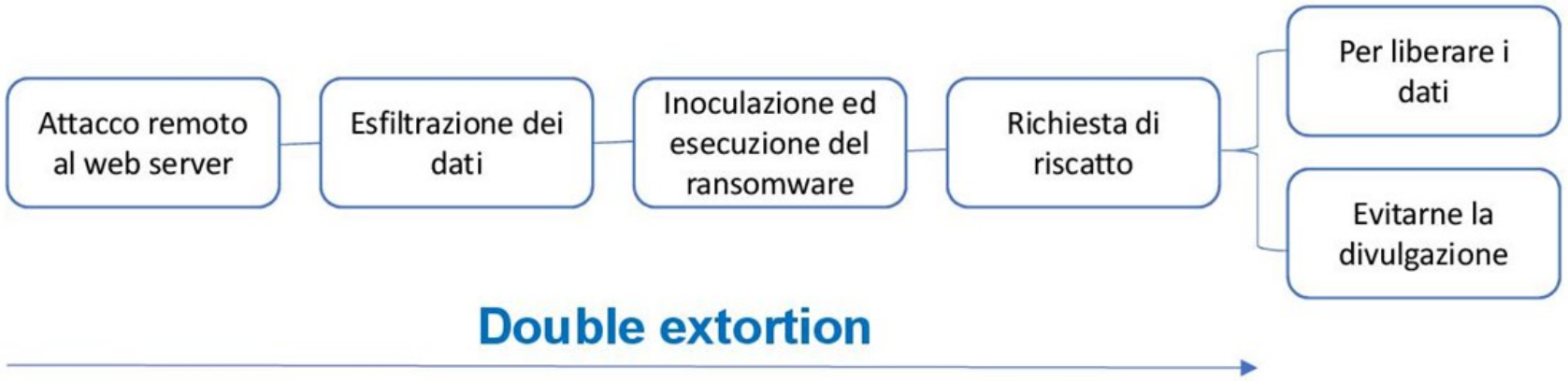




Ransomware + data leak

I ransomware sono noti per la cifratura dei dati e la conseguente richiesta di riscatto per liberarli. Di recente si è parlato di data leak causato dai ransomware Maze, NetWalker e Nefilim, ma la realtà è diversa, il Cert-AgID ha analizzato i sample nel dettaglio e dimostrato che questi ransomware non dispongono di alcuna componente in grado di esfiltrare i dati.

Maze: https://cert-agid.gov.it/news/il-ransomware-maze-chiude-era-davvero-in-grado-di-esfiltrare-dati/
NetWalker: https://cert-agid.gov.it/news/netwalker-il-ransomware-che-ha-beffato-lintera-community/
Nefilim: https://cert-agid.gov.it/news/il-ransomware-nefilim/



Fake breach e leak

Pubblicazione parziale di dati riciclati da altri leak. La procedura di estorsione resta identica a quella dei casi reali.
E' successo di recente al dipartimento di polizia di Minneapolis, ma a seguito delle analisi è emerso che 659/689 account erano parte del «dump» di LinkedIn del 2012.

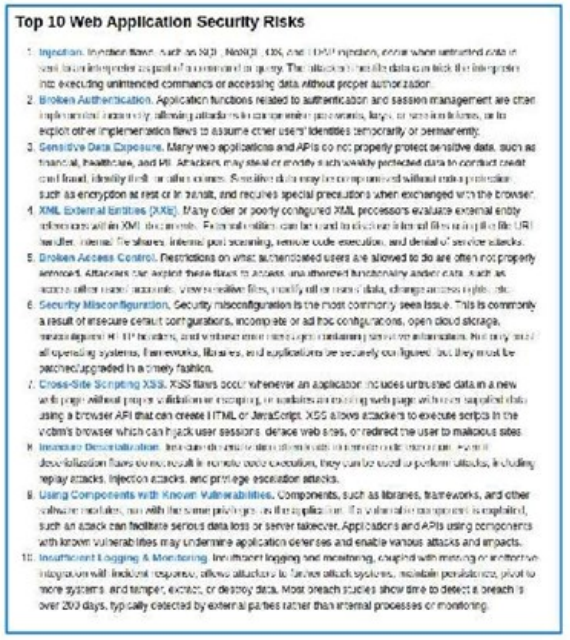
Web Application Attack

Le applicazioni web sono in grado di fornire risposte (informazioni) alle richieste dei visitatori grazie all'uso dei database. Se l'applicazione risulta essere vulnerabile l'intera base dati sarà esposta a rischio.

Gli attacchi più frequenti

Top 10 owasp: https://owasp.org/www-project-top-ten/

- SQL injection
- Sensitive Data Exposure
- Cross-Site Scripting (XSS)



Top 10 Web Application Security Risks

1. **Injection**. Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
2. **Broken Authentication**. Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
3. **Sensitive Data Exposure**. Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.
4. **XML External Entities (XXE)**. Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.
5. **Broken Access Control**. Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.
6. **Security Misconfiguration**. Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.
7. **Cross-Site Scripting XSS**. XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
8. **Insecure Deserialization**. Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.
9. **Using Components with Known Vulnerabilities**. Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
10. **Insufficient Logging & Monitoring**. Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

SQL injection

I dati passati in input da un utente malintenzionato possono interferire con le query che l'applicativo effettua al proprio database e di conseguenza restituire informazioni senza adeguata autorizzazione.

Un esempio

GET: https://insecure-hospital.com/**progetti**?**categoria**=covid
SQL: SELECT * FROM **progetti** WHERE **categoria** = 'covid' AND **visibile** = 1

visibile = 1 *mostra solo i progetti che possono essere visibili al pubblico*

GET: https://insecure-hospital.com/progetti?categoria=covid**'--**
SQL: SELECT * FROM progetti WHERE categoria = 'covid'--' AND visibile = 1

-- commento in SQL

SQL: SELECT * FROM progetti WHERE categoria = 'covid'
restituirà in output tutti i progetti, compresi quelli con flag visibile = 0

Sensitive Data Exposure

La priorità dello sviluppatore è quella di produrre un'applicazione funzionante, la sicurezza è (quasi) sempre pianificata come step successivo ed alla fine dimenticata, ignorata o fatta male a discapito della protezione dei dati e dei sui utenti.

Un esempio

- API token esposti nel codice sorgente
- Informazioni sensibili trasmesse o memorizzate in chiaro
- Credenziali deboli
- Cartelle annidate o sottodomini dimenticati

Cross-Site Scripting (XSS) Reflected o Persistent

Attacchi che sfruttano le debolezze di sicurezza insite nel codice dell'applicazione web per eseguire Javascript lato client.
L'input utente viene incluso nella pagina senza validarne il contenuto, il codice arbitrario viene eseguito consentendo all'aggressore di controllare il browser oppure, ove possibile, di utilizzare la sessione della vittima nel contesto dell'applicativo.

Un esempio

GET: https://insecure-hospital.com/search?text=covid
RES: Search: covid

GET: https://insecure-hospital.com/search?text=<script>alert(document.cookie)</script>
RES: PHPSESSID=qd66lO8djbtu823gr82c90vbt4

Proteggersi da questi attacchi?

Antivirus e protezioni perimetrali non sono sufficienti a contrastare le minacce appena descritte.

Quindi, cosa possiamo fare?

- Formare e sensibilizzare gli sviluppatori sui rischi legati a queste tipologie di vulnerabilità (rif. OWASP).
- Mantenere i framework aggiornati all'ultima release. Limitare l'uso di plugin di terze parti.
- Sfruttare al meglio i vantaggi della crittografia per memorizzare i dati nel DB e per la trasmissione delle informazioni.
- Effettuare periodicamente code review e VA/PT.
- Schedulare un processo di backup.

Come procedere se un attacco è andato a buon fine?

- Gestire l'incidente con il supporto di un team di esperti;
- Identificare ed analizzare la natura della violazione;
- Determinare la tipologia e la quantità dei dati eventualmente compromessi;
- Rilevare la possibilità di esfiltrazione;
- Predisporre un piano di remediation;
- Rilevare ed acquisire le evidenze informatiche;
- Estrapolare gli indicatori di compromissione (IoC);
- Utilizzare gli IoC per individuare ulteriori minacce della stessa tipologia;
- Valutare se e con chi condividere gli artefatti.

Grazie

vincenzocalabro.it