Data Professional Security Checklist

Identify the most common vulnerabilities and prevent a data breach

VALDAS MAKSIMAVIČIUS

2020 August

1. COMPROMISED AND WEAK CREDENTIALS

Compromised usernames, passwords or application tokens is one of the most common attack vectors. Around 80% of breaches happen due to insufficient authentication and authorization methods. The risk varies with the level of access it provides: regular user, system account, administrator

- ☐ 1. Avoid hard-coded passwords, keys, tokens (verify scripts, Jupyter notebooks, configuration files, etc.)
- ☐ 2. Always use standard authentication mechanisms (Client certificates, Windows based, Federated Azure Directory, Identity Server)
- ☐ 3. Whenever possible, prefer Federated (Active Directory) authentication
- ☐ 4. Enable Multi-Factor Authentication
- ☐ 5. Use dedicated system users for production environments with secrets knows only to admins
- ☐ 6. Store secrets, keys & certificates in password managers or key vaults
- ☐ 7. Don't store sensitive information in GIT (clean history if needed)
- ☐ 8. Don't share passwords via communication channels (Slack, Teams, email)
- ☐ 9. Ensure that password and account policy are implemented (strong password, rotation, soft lock-out, hard lock-out)
- ☐ 10. Ensure that secrets and passwords are stored in salted hash format
- ☐ 11. Ensure that user management events are logged (successful and failed user logins, password resets, password changes, account lockout, user registration)
- ☐ 12. Handle failed authentication scenarios securely
- ☐ 13. Ensure that administrative interfaces are appropriately locked down
- ☐ 14. Implement forgot password functionalities
- ☐ 15. Use finite lifetimes for generated tokens
- ☐ 16. Do not use master access tokens that provide direct owner access to services (e.g. storage keys, Event Hub)

2. MISSING OR POOR ENCRYPTION

Strong encryption must be applied to data at rest, in-motion, and where suitable, in-processing. Weak encryption can be just as bad as no encryption at all, as many legacy encryption algorithms can be cracked trivially using modern compute power.

- ☐ 1. Force all traffic through a secure HTTPS and TLS channel (verify Storages, App Services, data movement processes, etc.)
- ☐ 2. Verify X.509 certificates used to authenticate SSL/TLS
- ☐ 3. Verify the communication channel between your on-premises network and Azure (private/dedicated connection, IPSeC VPN, internet)
- ☐ 4. Enforce encryption and certificate validation in database connection strings
- ☐ 5. Enable Virtual Machine disk encryption
- ☐ 6. Enable data at rest encryption for all databases and storages
- ☐ 7. Enable backups encryption
- ☐ 8. Consider using Bring Your Own Key (BYOK) instead of cloud provider generated keys
- ☐ 9. Encrypt secrets, tokens and passwords in configuration files
- ☐ 10. Use only approved cryptographic hash functions (SHA-2 family)
- ☐ 11. Use strong encryption algorithms for encrypt functions
- ☐ 12. Add digital signature to critical securables (e.g Database securables such as a stored procedure, function, assembly, or trigger can be digitally signed)

3. MISCONFIGURATION

Misconfiguration happens when there is an error in system configuration. Misconfiguration has resulted in many high-profile breach events, with disastrous consequences. These are the issues that hackers monitor closely. Sounds scary, but one checkbox might lead to data exposure.

- ☐ 1. Disable public access to your storages and other resources (e.g. AWS S3, Azure Blob containers)
- ☐ 2. Ensure that only the required containers and blobs are given anonymous read access
- ☐ 3. Turn on firewall to control inbound and outbound traffic
- ☐ 4. Automate deployment in production not to miss important configuration details
- ☐ 5. Define and follow internal development standards
- ☐ 6. Train users on the risks associated with the cloud feature and good security practices
- ☐ 7. Disable tracing and debugging prior to deployment
- ☐ 8. Ensure log rotation and separation (log rotation is a way to limit the total size of the logs)
- ☐ 9. Ensure that only trusted origins are allowed if CORS is enabled (e.g. Storage, App services, etc.)
- ☐ 10. Separate administrative user sessions from regular usage (e.g., warnings, different color GUI when connecting to production, etc.)
- ☐ 11. Create budget alerts and resource creation controls
- ☐ 12. Enable auditing and logging on all the applications and storages
- ☐ 13. Enable auditing and logging on network, firewall and gateway levels
- ☐ 14. Ensure that audit and log files have restricted access

4. MALWARE, EXPLOITS AND UNPATCHED VULNERABILITIES

Malware injection attacks are done to take control of a user’s information. Hackers add malicious scripts, use unpatched servers, install exploits, encrypt data until a ransom is paid.
Overall, they seek to infect your endpoint or server.

- ☐ 1. Install endpoint protection on virtual machines and user PCs
- ☐ 2. Automate operating system and software patching
- ☐ 3. Enable security monitoring and alerts
- ☐ 4. Keep track of all your library dependencies and update regularly for new and legacy solutions running in production
- ☐ 5. Ensure that the deployed applications are run with least privileges
- ☐ 6. Don't give administrative privileges to any software
- ☐ 7. Implement Content Security Policy (CSP), disable inline JavaScript and iFraming (protection against XSS, MIME type sniffing, data exfiltration, click-jacking)
- ☐ 8. Enable threat detection on storages, databases and other services to seek anomalous activities indicating potential security threats
- ☐ 9. Enable cloud native DDoS protection

5. SENSITIVE DATA EXPOSURE

You should be especially careful with systems storing customer sensitive data. Under the GDPR, sensitive data is: data consisting of racial or ethnic originpolitical opinionsreligious or philosophical beliefstrade union membershipgenetic or biometric datadata concerning healthdata concerning a natural person's sex life or sexual orientation.

- ☐ 1. Store and process data only if you have a consent, right to process it under a specific purpose
- ☐ 2. Automate data lifecycle (data retention, masking, etc.)
- ☐ 3. Ensure sensitive information is not cached in other places than specified by design (e.g. on the browser cache)
- ☐ 4. Mask sensitive data displayed on the user screen (unless there is a lawful reason)
- ☐ 5. Implement dynamic data masking to limit sensitive data exposure to non privileged users
- ☐ 6. Ensure that sensitive data in database columns is encrypted
- ☐ 7. Perform security modeling and use Business Units/Teams where required
- ☐ 8. Ensure an application does not log sensitive user data
- ☐ 9. Identify sensitive entities and implement change auditing
- ☐ 10. Implement Row Level Security (RLS) to prevent tenants from accessing each other's data

6. PHISHING

Phishing is a social engineering technique where the target is contacted by email, telephone or text message by someone who is posing to be a legitimate colleague or institution to trick them into providing sensitive data, credentials or personally identifiable information (PII).

Phishing – the human element always has, and likely always will be, a key contributing factor in information security failure.

- ☐ 1. Train users on the risks associated with the cloud features and potential social engineering scenarios
- ☐ 2. Do not click any links or download any attachments in the suspicious email
- ☐ 3. Never give personal information, system details, user names, password or other sensitive details over the phone
- ☐ 4. Install security protection software on your devices (laptop, phone, etc.)
- ☐ 5. When in doubt – 1. reach out to your company; 2. investigate the organization you received the email from to determine if it is a phishing scam or not.

7. MALICIOUS INSIDERS AND MISUSE

A malicious insider is an employee who exposes private information and/or exploits vulnerabilities. Since these users are legitimate, it can be more difficult to detect these types of attacks than most others.

- ☐ 1. Enable fine-grained access management to Azure Subscription using RBAC
- ☐ 2. Ensure that the system has inbuilt defenses against misuse
- ☐ 3. Ensure that only the minimum required services/features are enabled
- ☐ 4. Do not expose security details in error messages (e.g. server names, connection strings, usernames, passwords, SQL procedures)
- ☐ 5. Use the lowest possible privileges – Principle of Least Privilege (PoLP)
- ☐ 6. Keep an eye out for dissatisfied employees

8. THIRD AND FOURTH-PARTY COMPONENTS AND VENDORS

There are many interconnected systems, both within, outside and across organizations. This complex set of relationships has the potential to be exploited by attackers.

- ☐ 1. Disable Azure/AWS/GCP marketplace by default
- ☐ 2. Access third-party components from trusted sources only
- ☐ 3. Verify approved versions of common JavaScript/Python/R/Java/else libraries that do not contain known security flaws
- ☐ 4. Keep up to date with the latest updates and new releases from vendors
- ☐ 5. Involve legal and procurement teams in case of new licences
- ☐ 6. Don't expose sensitive information to your vendors (unless approved by your organization)
- ☐ 7. Validate licence details before using trial/demo products

ABOUT THE AUTHOR

Hi there!

I am a software architect specializing in data analytics and cloud computing with ten years of experience. I have been using Azure Cloud components since 2014.

For the last five years, I have been leading Data Engineering teams using the latest Azure Data and AI services. I worked on Data Lake and Data Science platform implementations for various sectors in the Nordics.

I enjoy sharing my lessons learned at conferences and meetups. I am a founder of Vilnius Microsoft Data Platform Meetup and a frequent speaker at IT conferences.

Let’s get in touch!
Valdas Maksimavičius
valdas@maksimavicius.eu

https://www.valdas.blog
https://www.dataplatformschool.com