Top 15 Cyber Security Controls

Version 1.0 draft #2

September 2024

Introduction

Description

This is a prioritized list of primarily technical cyber security controls that was deduced by analyzing known cyber attacks, that took place over the last decade around the world; attacks which very likely would have been deflected if these controls were in place. This is original research, and it was not derived from any existing framework, guide, etc. Source materials are published in the project's GitHub repository.

This work began in 2023 out of pure curiosity and a desire to answer a question: "What attack tactics & techniques (T&T's) are the most common *today*, and how does their heatmap look like?" and, subsequently: "What is the minimum number of controls that should be put in place to defend against these T&T's?" Hundreds of attacks were mapped to MITRE ATT&CK, leading to a map of 500+ mitigation techniques, that were grouped, summarized, and finally compressed to just "Top 15" unique controls. Additional details and notes were added to each control to ensure that critical requirements are not missed.

Relation to the existing security frameworks

Established frameworks, such as ISO 27001, NIST CSF and CIS Controls, provide a comprehensive approach to managing security in an organization and should be used as main sources of knowledge. However, these frameworks require a security manager to spend quite a few hours (days, weeks) to conduct a risk assessment and figure out which controls are applicable and, sometimes, what do they mean... Unfortunately, this exercise often ends up being an educated guess, and, as the result, organizations tend to implement complex controls first while missing the basics.

"Top 15" complements existing frameworks (standards, guides) by providing an opportunity to discover major security gaps in your company and prioritize controls implementation. The main idea here is that it should not take more than 15 minutes.

Who can use this and how?

- **CISO, Head of Security, Security Managers, Consultants** — as described above.
- **Pentesters** — focus on the statistically most problematic parts of the attack surface.
- **Vendors, Sales** — leverage real-world data when positioning your products before clients.

There is an accompanying list of vendors and their product available in the project's GitHub repository in a file named **"Top15 - tools.xlsx"** The work on this document is still in progress, but one can get an idea of which products could satisfy the controls' requirements.

Acknowledgements

TBD

People

Organizations

MITRE (ATT&CK)

Palo Alto (Unit 42)

CISA.gov (Advisories)

Port Swigger (Daily Swig)

Trend Micro (Research)

Mandiant (Threat Intel)

The Hacker News

Bleeping Computer

Dark Reading

Contents

Control #1: Utilize Privileged/User Account Management solution with continuous audit.

1.1. Follow the principle of least privilege.
1.2. Follow the Just in Time/Just Enough Administration (JIT/JEA) practices.
1.3. Conduct periodic access reviews for human and service (applications) identities.
1.4. Utilize phish-resistant multi-factor authentication (MFA).
1.5. Accounts that cannot be protected by MFA must have complex, unique passwords across all systems on the network; passwords must be rotated on a regular basis.
1.6. Utilize Local Administrator Password Solution (LAPS).

Related mitigation techniques: M1026, M1018

Rationale:

1.1. The importance of the principle of least privilege (PoLP) is paramount: this control alone is responsible for ~80% of the whole defense — and it's one of the most overlooked controls at the same time. This principle is quite a very broad concept which prescribes that any human or programmatic entity should only have access to specific data, resources and applications that are required by that entity to do their job. If an attacked compromises that entity's account, the blast radius of the attack will be limited.
1.2. Obtaining administrator privileges is one of the most desirable outcomes for attackers, as it often gives keys to the kingdom — and its crown jewels. Simply, Just in Time (JIT) access is a feature that removes all administrative privileges from administrators' accounts by default, and only assigns them back upon explicit request subject to additional approval and time limits. Even if an attacker compromises such account, they get no administrative access by default.
Just Enough Administration (JEA) constrains the administrators even further by specifying which cmdlets, functions, and external commands they can run.
1.3. There is a known problem with privileges: they tend to accumulate over time as users change roles, get promoted, assigned to new projects and groups, etc. It is called "privilege creep", and it's a direct and the most common violation of the PoLP (a). Sometimes privileges are being assigned by mistake, opening holes in the defense. Periodic access reviews help resolve this problem, and automated tools are of great help here.
1.4. Multi-factor authentication concept is simple: one should not only know a username and a password but also have something specific (e.g., provide unique one-time code generated by an application) or (rarely) be someone specific (e.g., provide fingerprints). It was supposed to be a barrier for phishing attacks, but ironically attackers have found a way to obtain one-time codes or authentication cookies by… phishing attacks!
Phish-resistant MFA consists of a few security measures that cannot be phished by design: physical tokens, digital certificates (sometimes stored on tokens), and device-based

authentication (such as Windows Hello). To be used by an attacker, these must be physically stolen from a user; additional layer of encryption with a PIN-code or password usually mitigates this threat.

1.5. Service (application) accounts are the ones that cannot be protected by MFA: they don't have smartphone applications or fingerprints to present. The only way to keep them adequately secure is to utilize extra-long passwords (e.g., 64 alpha-numeric characters) and rotate the passwords periodically (e.g., every 180 days).

1.6. There are multiple problems with static Administrator account on users' workstations. For example, if a "golden image" is used to prepare a standard OS installation, it is very likely that all workstations will have Administrator account with the same password. Also, to save time, IT support may set up the same password to simplify administration. If the password is compromised on one machine, it's compromised on all of them.
Local Administrator Password Solution (LAPS) is designed to periodically change the local Administrator account password on each machine to a unique complex value.

Control #2: Utilize anti-malware solution with behaviour analytics.

2.1. Solution must provide functionality at least equal to Microsoft Defender for Endpoint Attack Surface Reduction (ASR).

Related mitigation techniques: M1040

Rationale:

2.1. Anti-malware solutions have been "must haves" for a long time: when everything else has failed, they are likely the last technical line of defense. Modern solutions may still use signature (hash) based detections, but most of their value could be found in behaviour analysis that operates in tandem with a cloud-based "hive mind". Together they inspect every action of the potential malware under a microscope and make conclusions about the level of danger.
There are many anti-malware solutions on the market, but they are not equal. The baseline is Microsoft Defender for Endpoint's Attack Surface Reduction (ASR), a set of rules that can constrain software-based risky behaviors and prevent infiltration. This functionality is not unique and may be found in other solutions, but it should be present in full.

Control #3: Apply current operation systems hardening best practices.

3.1. Consider using CIS Benchmarks.

Related mitigation techniques: M1028

Rationale:

3.1. By default, many operating systems, especially end-user ones, are configured to provide most comfortable user experience but often fall short of security. Large number of enabled services and preinstalled applications and default configurations that impose the least restrictions on the user are greatly appreciated by the latter but open a number of holes in the defense. Services and applications tend to have vulnerabilities that need to be patched even if the user never uses them; if a workstation is compromised, the attacker will have the same restrictions as the user — the least ones.
CIS Benchmarks has become a standard in OS hardening, which includes *a lot* of extremely detailed recommendations for default configuration changes. CIS Benchmarks are used by many vendors in their solutions for on-prem and cloud environments.

Control #4: Utilize a Network Intrusion Prevention System.

Note: for cloud-only environments utilize native firewalls where applicable.

4.1. NIPS should be able to inspect SSL/TLS traffic, recognize known attack/malware traffic patterns, malicious domains and IPs.
4.2. Segment the network and block unnecessary traffic between segments.
4.3. Use remote desktop gateways when applicable and if VPN is not possible.

Related mitigation techniques: M1031

Rationale:

4.1. Widespread introduction of traffic encryption between end-user applications and cloud services increased security of the data on the one hand but created additional problems for the defenders on the other. How can we recognize if the traffic is not being sent by a piece of malware or there is no data exfiltration takes place? With a few exceptions for particularly security-aware applications that check their server certificates, NIPS should be able to inspect all encrypted traffic. Some vendors provide an endpoint client or extension that can intercept not-yet-encrypted traffic and send it for inspection.
There are more ways to prevent certain attacks without decryption. One is to stop communications with a known malicious domains and IP addresses, and another one is to block known attack/malware traffic by recognizing their patterns. Both ways leverage

databases which are usually maintained by vendors or third parties.

4.2. Network segmentation with traffic filtering between segments is essentially an extension of the principle of least privilege to the network: if one should not use services on another subnet, they should not have access to it. This also helps to reduce the attack surface in case of ransomware/worm scenario.
Ideally, all network traffic between all endpoints, including servers and other network hardware, should be filtered on a firewall; however, this is quite cumbersome.

4.3. Under no circumstances remote desktop services should be published directly on the Internet. RDP has a few known vulnerabilities, it is subject to brute-force password attacks, and attackers will be happy to see it available without any additional protection. Unless VPN is not an option for some cases, a remote desktop gateway put into a DMZ should be utilized.

Control #5: Utilize an execution prevention solution.

Note: may be embedded in Control #2: Behavior Prevention on Endpoint.

5.1. Users should not be allowed to run system utilities (reg, mshta, cmstp, InstallUtil, odbcconf, hh, certutil, *.cpl files) and scripting interpreters (VB, Python, Powershell) not required by their jobs.

5.2. Block unknown DLLs.

5.3. Ensure that accessibility features are protected from change or blocked if not required.

Related mitigation techniques: M1038

Rationale:

5.1. Execution prevention, also known as application whitelisting or application control, is extremely powerful tool for blocking unknown applications — and therefore threats they impose. However, attackers have learned to 'live off the land' and utilize existing OS functionality to bypass security controls and/or elevate privileges. For example, certutil can be used to download malicious files directly while it is blocked in a browser, while InstallUtil can be used to proxy code execution. Similar to the latter, unattended scripting interpreters allow nearly any code to be executed while source code, being a simple text which could be obfuscated, is less likely to be analyzed properly by antimalware engines.

5.2. DLL files, while being less suspicious compared to executables, can still contain malicious code and attackers use them all the time: a simple call via rundll32 is enough to make bad things running.

5.3. Another example of attackers living off the land is their technique(s) that replaces legitimate accessibility tools (e.g., On-Screen Keyboard, Magnifier, Narrator, etc.) with a malicious executable, which then used for privilege escalation (up to SYSTEM) and persistence.

Control #6: Conduct periodic user security awareness training.

6.1. At least, cover most common attack techniques e.g., spear phishing, MFA bombing, etc.

Related mitigation techniques: M1017

Rationale:

6.1. Approximately 10% of trained users still can be tricked to open a link in a phishing email and submit their credentials on a shady website that looks exactly like their company's web portal. While it sounds helpless ("nothing can be done about it"), it worth knowing that *3 times more* users will do the same without training.
Experts still debate whether or not phishing simulation should be used, as sometimes test messages' designers may cross the line and even end up in legal disputes, there are very few doubts, if any, about the education itself.

Control #7: Disable or remove applications, services, features and add-ins that are not explicitly needed.

Related mitigation techniques: M1042

Rationale:

7.0. One of the easiest ways to have fewer vulnerabilities to patch is to have fewer software in the first place. By default, operating systems, especially desktop ones, tend to have a lot of preinstalled applications and running services as well as enabled features, while many of them may be never used but often abused by attackers (e.g., Remote Desktop, WinRM service, autorun feature). Reducing the attack surface by removing unnecessary software together with their features and add-ins is a great extension to Control #3 (Operating System hardening) and #5 (Execution Prevention).
Boundaries of this controls, however, are not limited to common day-to-day software. For example, scripting languages such as PHP have a built-in way to disable just the certain dangerous functions, which have direct access to the OS and the file system, while the rest of the software will serve its clients.

Control #8: Filter web traffic on a firewall (proxy) and/or on endpoint.

Related mitigation techniques: M1021

Rationale:

8.0. A lot of malicious content could enter the endpoint via a browser; therefore, it is important to filter as much web traffic as possible. While filtering can occur early on a firewall or a dedicated proxy (see Control #4, Network Intrusion Prevention), due to traffic encryption and certificate pinning discussed in [4.1], it is rarely possible to decrypt web traffic without breaking certain cloud applications; it is still possible though to conduct category-based DNS filtering. Under those circumstances it may be more effective to implement additional in-browser web traffic filtering as traffic is already decrypted.
This functionality may be embedded in solutions for Control #2 (Behavior Prevention on Endpoint).

Control #9: Filter malicious content on the email gateway.

9.1. Configure Sender Policy Framework (SPF) and DomainKeys Identified Mail (DKIM).
9.2. Configure Domain-based Message Authentication Reporting and Conformance (DMARC).
9.3. Configure email gateway to validate incoming emails against SPF and DKIM and follow DMARC where applicable.

Related mitigation techniques: M1054

Rationale:

9.1. An SPF record for a domain provides a list of IP addresses (as well as subnets or domain names that could be resolved to IPs), from which emails from this domain could be sent, together with instructions (accept, discard, soft fail, do nothing) on what to do. Email gateways that receive emails from a domain with an SPF record should verify that the envelope originated from an allowed IP address and follow the instructions.
DKIM is a combination of an email digital signature mechanism and a DNS record that contains corresponding public keys. Similar to SPF, email gateways should verify digital signatures of incoming emails, but certain actions are not prescribed in the record itself.
9.2. DMARC record instructs the email gateway on how to validate incoming emails against both SPF and DKIM and to verify so-called “alignment”, an additional check that is not covered by SPF. Additionally, DMARC allows email administrators to debug the mail flow by providing a report email address.
9.3. While requirements [9.1] and [9.2] are focused on providing valuable information to email gateways that receive envelopes from your organization, incoming mail should be validated the same way to protect users from threats originating from spoofed domains.

Control #10: Encrypt sensitive information at rest and in transit.

10.1. Enable Secure Boot and configure full disk encryption (FDE) for local and removable drives. If Trusted Platform Module (TPM) is used, ensure a password/PIN is also used where applicable.
10.2. Encrypt network traffic containing sensitive information (e.g., Kerberos, audit logs, etc.)
10.3. Utilize strong cyphers (e.g., AES-128/256, Elliptic Curves-based) with adequate keys.

Related mitigation techniques: M1041

Rationale:

10.1. If a laptop or a flash drive with sensitive information is lost or stolen, it is strong encryption that stands between the information and the prying, potentially adversarial eyes. TPM is (was) supposed to securely store the encryption key, but current hardware implementations transmit the key in plain text during the boot phase, and recent successful attacks show that the key must be protected with a password or at least a PIN.
10.2. Sensitive information that is secure while residing on an encrypted storage, may become vulnerable during transit due to network sniffing (eavesdropping). Sensitive network traffic does not only include usual files, but also authentication traffic (e.g., Kerberos), audit events traffic (if transferred to a SIEM), and other application traffic.
10.3. It is important not only to encrypt sensitive information but to do it properly. Outdated cyphers such as DES, RC4, which could be easily broken, have no good use in a secure environment but often found in default configurations. The definition of an adequate key depends on the cypher: it should be complex and long enough for symmetric cyphers (e.g., AES), but this largely does not apply to elliptic key cryptography (ECC) keys, which could be 256-bit long but provide sufficient protection due to the nature of the algorithm.

Control #11: Conduct regular vulnerability scans of the network infrastructure and the endpoints, update software and firmware.

Related mitigation techniques: M1051

Rationale:

11.0 This control can compete with the principle of least privilege (Control #1.1) for being the most overlooked. Recent studies show that unpatched vulnerabilities were exploited at some point in 60% of breaches, while 30% of vulnerabilities remain unpatched *for over a year*. At the same time, attackers can develop exploits as quick as in 48 hours. Therefore, it is important to have solution that automatically scans the network infrastructure and the endpoints, including the remote ones, and provide automated analysis and patching capabilities.
Keep in mind that updates/patches should be tested on small subsets before wide implementation.

Control #12: Conduct regular Active Directory vulnerability assessments, implement recommended fixes.

Related mitigation techniques: M1015

Rationale:

12.0 Common OS and software vulnerabilities may be neglected for different reasons, but it is usually done knowingly; on the contrary, there is a very little discussion of Active Directory hardening. Default configuration of an AD domain contains quite a few vulnerabilities and weaknesses, and it could lead to a full domain compromise relatively quickly. While there are usually tens of potential problems, the most common fixes that could be applied as soon as possible are:

- Block LLMNR and NetBIOS traffic.
- Enable SMB Signing.

Control #13: Establish secure software development practices.

13.1 Conduct application developers-specific training that should include knowledge of OWASP's **Top 10** and **Application Security Verification Standard (ASVS)**, and adhere to these standards.

13.2 Conduct threat modelling, weakness and vulnerability scans of the source code (SAST), compiled application security testing (DAST), third-party libraries (SCA), used container images and Infrastructure-as-Code (IAC) templates.

Related mitigation techniques: M1013

Rationale:

13.1 Uncountable number of applications and their corresponding back-end APIs are published on the Web, and this number grows daily. Many of these applications are released as 'minimum viable product', and some minimal security in such products exists only thanks to frameworks that were used for development; however, this is not enough in long term.
OWASP's Top 10, per their own definition, is the reference standard for the most critical web application security risks, while ASVS provides a basis for testing web application technical security controls and provides developers with a list of requirements for secure development. To build and keep applications secure, it is important to train developers on these standards and to verify that they are continuously followed.

13.2 Aside from the governance processes described in [13.1], there are practical exercises that should be embedded into the development pipeline:

- Application threat modelling is a process of building a list or a map of potential threats, prioritizing them and mapping corresponding security controls.
- Weakness and vulnerability scans of the source code are done by 'Static Application Security Testing (SAST)', while the compiled application is examined by 'Dynamic Application Security Testing (DAST)'.

- Third-party open-source libraries often contain vulnerabilities, and a process to detect them, as well as to verify licensing compliance, is known as 'Software Composition Analysis (SCA)'.
- Preconfigured container images (e.g., used in Docker) also often contain vulnerabilities and must be regularly updated; there is no specific abbreviated name for this process, but it could be referred to as 'Container Security [check]'.
- Infrastructure-as-Code (IAC) templates are used to create standardized cloud environment by defining exact components and their configuration in a simple text file, while automatic compliance check helps with maintaining defined configuration. IAC templates may include vulnerable components (e.g., outdated images) and must be checked regularly.

Control #14: Implement Data Loss Prevention.

14.1 The solution must cover endpoints.
14.2 Restrict access to removable storage devices and online storage.

Related mitigation techniques: M1057

Rationale:

14.1 There are many DLP solutions, but some of them is cloud-only and therefore completely miss on what happens on remote endpoints and how potentially sensitive data is handled there. A locally installed agent should close this gap.
14.2 Removable storage devices (USB flash drives) and online storage (Google Drive, Microsoft OneDrive, Dropbox, etc.) are the easiest and most widely used ways to exfiltrate data. Restricting access to them is possible even without a DLP solution: USB drives can blocked using group policy objects (GPO) or additional anti-malware (Control #2) functionality, while online storage could be filtered on the firewall (Control #8).

Control #15: Perform regular data backups.

15.1 Store at least one read-only copy in a remote location.
15.2 Test backups periodically.
15.3 Encrypt backups.
15.4 Include Active Directory backup and/or rollback solutions as a separate option.

Related mitigation techniques: M1053

Rationale:

15.1 When things go wrong, backups are the last resort to restore data and systems, but what if things go *seriously* wrong? Storing at least one copy of backups in a remote location, such as in a datacenter on the opposite side of the country, could be of help in this dire situation. With major cloud providers, any required storage space is just a few clicks away.

It also important to make sure that the process cannot overwrite existing remote backups: prefer 'pulling' backups over 'pushing' them, if possible.

15.2 Backups are not true backups until they have been tested. Testing should normally include single file/database recovery and single machine recovery, either as a virtual machine or on a bare-metal server as applicable. It can also extend to testing different scenarios for disaster recovery for those who would like to be fully prepared.

15.3 Many breaches occurred due to poorly protected backup copies stored in plaintext. Encryption of backups, both in transit and at rest, should be a mandatory minimum, but ideally backups should be protected with the same rigor as the original data.

15.4 Although it is relatively simple to backup an Active Directory (AD) domain information by backing up the whole domain controller (full system backup) or by capturing system state (custom backup), restoration of the whole systems may not be as easy or even not possible at all. There are dedicated solutions that focus on AD only, and they can restore it back much faster. Other solutions offer even better option to rollback any changes on-the-fly, while defending the domain from malicious changes, which is usually done by preconfigured policies.

Artificial Intelligence usage disclosure

No Artificial Intelligence, including Generative AI, LLMs, neural networks and other buzzwords were used during the research process and in the making of this document.

License

Sincerely,
Valeriy 'Val' Shevtsov

Change Log

Date	Version	Change
2024-11-18	1.0 draft 2	Added rationale for each of the sub requirements Renamed sections: descriptions became titles
2024-09-05	1.0 draft 1	Initial draft; timed to my dad's 0b1000000'th birthday