Communications Security Establishment
Centre de la sécurité des télécommunications

CANADIAN CENTRE FOR CYBER SECURITY

Strategies for Protecting Web Application Systems Against Credential Stuffing Attacks

PRACTITIONER

TLP:WHITE

Canada

FOREWORD

ITSP.30.035 Strategies for Protecting Web Application Systems Against Credential Stuffing Attacks is an UNCLASSIFIED publication issued under the authority of the Head of the Canadian Centre for Cyber Security (Cyber Centre).

For more information, phone or email our Contact Centre:

Contact Centre

contact@cyber.gc.ca
(613) 949-7048 or 1-833-CYBER-88

EFFECTIVE DATE

This publication takes effect on January 17, 2022.

REVISION HISTORY

Revision	Amendments	Date
1	First release.	January 17, 2022

ISBN 978-0-660-40252-9
CAT D97-3/30-035-2021E-PDF

OVERVIEW

This document provides web application and system administrators with security recommendations to protect web application systems against credential stuffing attacks.

Canadians rely on web-based applications to interact with critical services. Both public and private organizations increasingly deploy web applications to provide secure, flexible, and resilient online service platforms. Securing these operations against authentication-related attacks is crucial to protecting the confidentiality and integrity of systems and information.

This document recommends technical and non-technical security controls that you can apply to safeguard your organization's web services against credential stuffing attacks. While this document may directly address Government of Canada (GC) systems, the guidance in this document is applicable for non-GC organizations.

TABLE OF CONTENTS

LIST OF TABLES

1 INTRODUCTION

This document discusses common password authentication threats but focuses primarily on credential stuffing attacks. It recommends some technical and non-technical security controls that you can implement to prevent and mitigate credential stuffing attacks. While this document addresses GC systems, non-GC organizations can also apply the recommendations.

In today's connected world, web applications provide a cost-effective and nimble infrastructure to rapidly build and deliver services. Organizations depend on applications to deliver critical services to their customers. The need for a safe and secure authentication framework is essential, especially in more complex deployments, such as highly sensitive or critical systems. For example, GC departments use web applications to deliver a diverse set of services. Often, users and employees are required to authenticate to access these applications. To establish trust and confidence in the system, the authentication framework must accurately identify legitimate users and repel adversary password attacks.

The increase in reported cases of corporate data breaches has led to growing cases of credential stuffing incidents. Millions of credential attacks occur daily, targeting web application systems on the Internet. Threat actors target user accounts using leaked or stolen credentials to gain unauthorized access to accounts. Consequently, you need to review your web application's authentication architecture to secure it against password-based attacks.

We recommend reviewing the following documents for more information on how to secure your web application services:

- *ITSP.30.031 v3 User Authentication Guidance for Information Technology Systems* [1][1]
- *National Institute of Standards and Technology (NIST) Special Publication (SP) 800-63B Digital Identity Guidelines: Authentication and Lifecycle Management* [2]
- *NIST SP 800-95 Guide to Secure Web Services* [3]
- *Open Web Application Security Project (OWASP) Application Security Verification Standard 4.0* [4]

1.1 SCOPE

This document addresses security control protections against credential stuffing attacks. Although credential stuffing attacks are closely related to brute force, password spraying, or other forms of password-related attacks, the suggested controls are primarily tailored to protect against credential stuffing.

This document provides guidance for web applications that hold data with a low or moderate level of sensitivity; in the GC context, this refers to unclassified, Protected A, or Protected B data. This document does not apply to web applications that hold highly sensitive data (Protected C data) or classified systems that hold sensitive data of national interest.

1.2 AUTHENTICATION AND IT SECURITY RISK MANAGEMENT PROCESS

Ensuring the integrity of data and system processes is a fundamental objective when building and deploying secure web applications. The requirement to accurately authenticate and authorize user service requests is closely linked to the

[1] Numbers in square brackets refer to references listed in the Supporting Content section of this document.

system's confidentiality and integrity objectives. You should align your security processes with existing IT security risk management frameworks to preserve the integrity of your web application system. Identification and authentication mechanisms establish security procedures for identifying and validating that users are who they claim to be.

ITSP.30.031 v3 [1] provides information on recommended security controls for GC web application systems. Additionally, NIST provides guidance on how to develop and deploy secure authentication architectures in the following publications:

- *NIST SP 800-63-3 Digital Identity Guidelines* [5]
- *NIST SP 800-63A Digital Identity Guidelines: Enrollment and Identity Proofing* [6]
- NIST SP 800-63B [2]
- *NIST SP 800-63C Digital Identity Guidelines: Federation and Assertions* [7]

1.2.1 IDENTIFICATION AND AUTHENTICATION

Identification and authentication refer to the family of security controls that enables an information system to uniquely identify and authenticate users. These include authentication activities related to guiding policies and procedures, identifying users uniquely, managing identifiers, managing authenticators, managing feedback messages, proofing identities, and re-authenticating.

For a detailed description of the identification and authentication control activities, refer to the section on identification and authentication in Annex 3a of *ITSG-33 IT Security Risk Management: A Lifecycle Approach* [8].

1.2.2 IDENTITY PROOFING

Identity proofing refers to the process of collecting, validating, and verifying a user's identity information to establish credentials for accessing a system. Identity proofing involves activities that confirm a user is who they claim to be.
The components of the identity proofing process can vary depending on the security profile of the web application. NIST SP 800-63A [6] describes three options available to identity proof and enroll users. Each option lists different sets of requirements based on the system's risk profile. Within most web application systems, these activities include user identification, registration, and validation processes. To maintain the integrity of a web application, system owners and administrators must implement requirements that align with their systems' risk profiles. Details of selecting a web application's risk profile are beyond the scope of this document.

1.2.3 CREDENTIAL STUFFING

Credential stuffing is an attack technique by which threat actors use leaked or stolen credential information to access user account profiles on other web applications. In other words, threat actors use access credentials they have acquired elsewhere – usually from prior data breaches – to 'stuff' them into the credentials for your system and log in as an authorized user. Credential stuffing exploits the user's behaviour of reusing login credentials. Please refer to section 2.1.3 of this document for more information on credential stuffing attacks.

1.3 AUTHENTICATION

In this section, we review forms of authentication and factors of authentication.

Digital authentication is a process involving a user presenting and claiming an identity tied to specific credentials. Digital authentication systems rely heavily on knowledge-based authentication systems. Often, these are usernames and passwords. However, more online platforms now support multi-factor authentication (MFA), such as using hardware tokens as an additional factor. The goal of the authentication process is to validate legitimate requests while rejecting malicious authentication attempts.

1.3.1 AUTHENTICATION FACTORS

The authentication factor refers to the property of the parameter being used to validate the user identity during the authentication process. Table 1 describes the different authentication factors and their characteristics and provides some examples.

Table 1: Authentication Factors

Characteristic	Description	Examples
Something a user knows	Information that only the legitimate user should know	Password, PIN
Something a user has	A physical object that only a legitimate user processes and controls	Hardware token, mobile phone, device signature, private key, bank card
Something a user is or does	A physical attribute that is unique to each user	Fingerprint, retina, face, voice, or behaviour pattern such as typing speed

MFA requires at least two authentication factors from different characteristic levels (e.g. something a user knows and something a user is). Generally, MFA improves the security of the authentication process. Though still widely used, MFA codes distributed via traditional telephony technologies (short message service [SMS] texts and voice call) can be easily intercepted. Threat actors can use basic attack techniques, such as subscriber identity module (SIM) swapping, Signaling System 7 (SS7) communication intercept attacks, and phishing, to gain access to the authentication code.

Using authentication factors with the same characteristics does not represent MFA. For example, a user password and a security question are considered to possess the same characteristic, as they are something a user knows. Two factors with the same characteristic can be compromised the same way. For example, a phishing email or key logger can pick up a password, a security question, or a PIN. For a detailed listing of token types and their description, refer to ITSP.30.031 v3 [1].

2 PASSWORD AUTHENTICATION ATTACKS

Password authentication attacks occur when threat actors launch offline or online attempts to gain unauthorized access to a user account using legitimate credentials. In offline attacks, threat actors use tools to crack user passwords. For online attacks, they exploit web systems' weak authentication controls to gain unauthorized access. Threat actors can use multiple passwords to target a specific user account or use a password against an entire user database.

Many web applications require email addresses as usernames on their platform. Over time, this may encourage users to reuse usernames and passwords across different platforms. If a credential pair used on a web platform is compromised, threat actors may be able to use the pair on other platforms.

2.1 PASSWORD ATTACK METHODS

In this section, we review some of the common methods that threat actors use to carry out password authentication attacks.

2.1.1 BRUTE FORCE ATTACKS

In a brute force attack, a threat actor cycles through all possible character combinations and attempts to log in until they find a match. Threat actors can launch brute force attacks offline or against online systems. If attacking online applications, a threat actor targets an account on web systems that do not have rate-limiting or account lockout protections.

2.1.2 DICTIONARY ATTACKS

A threat actor uses a list of common passwords and attempts to log in until they find a match. Typically, threat actors use passwords composed of dictionary words.

2.1.3 PASSWORD SPRAYING ATTACKS

Password spraying is a variant of an online brute force attack. It involves the use of a small set of common passwords to log into several user accounts. The threat actor targets multiple user accounts to evade account lockout controls and rate-limiting protections.

2.1.4 CREDENTIAL STUFFING ATTACKS

A credential stuffing attack occurs when threat actors use a list of leaked combinations of usernames (often email addresses) and passwords to authenticate to a web application. In other words, threat actors use access credentials they have acquired elsewhere to 'stuff' them into the credentials for your system, allowing them to sign as an authorized user.

Credential stuffing exploits users' tendency to reuse login credentials. Hackers use automated bots to launch sequential login attempts to validate a credential list and identify successful combinations. Threat actors can also launch credential stuffing attacks using human hacking farms managed by criminal gangs to mimic typical user behaviour. Typically, threat actors use credentials obtained from data breaches or acquired from underground dark communities. In advanced stuffing attacks, credentials are paired with specific user data like date of birth information [9].

Defending against credential stuffing attacks can be quite daunting. A web application is vulnerable not because of a security breach on its infrastructure but rather because of its users reusing login credentials. System security administrators may detect an attack in progress by monitoring and running analytics on failed authentication login records. In section 3, we discuss more security control mechanisms that you can use to mitigate and protect against credential stuffing attacks.

2.2 COMMON ACCOUNTS TARGETED

Credential stuffing attacks target the following account types:

- **User accounts**: These are end user accounts that are typically accessed using usernames (often email addresses) and password pairs. Most compromised credential databases consist of email addresses and password pairs.
- **System accounts**: These are accounts associated with processes, services, or system devices. The use of weak passwords on Internet of Things (IoT) devices makes them an attractive target. Threat actors take over vulnerable IoT devices and deploy them as malicious bots. System accounts may also include accounts on firewall or networking devices and those linked to several web application services such as database services.
- **Application programming interfaces (APIs):** The proliferation of web services led to the growth in the use of API interfaces. Threat actors target API keys because account protection mechanisms such as MFA tokens are not usually enabled.
- **Federated identity accounts**: Threat actors place a premium on federated identities, such as single sign-on (SSO) credentials, because they facilitate easy access to multiple services. Many cloud deployments involve hybrid architectures, and secure communication between these systems requires identity federation. Successful authentication and authorized requests may require sharing authentication tokens over the Internet. Sophisticated threat actors can collect encrypted network traffic data and use advanced cryptographic techniques to extract token credentials.

2.3 COMMON COMPROMISED CREDENTIAL SOURCES

Threat actors depend on a variety of sources to harvest compromised credentials. They may use data used from the following sources:

- **Data breaches:** When threat actors breach corporate systems, they steal credential databases and post the data publicly. Your organization can monitor public credential breach data to identify accounts impacted on your platform. Please confirm the legal implications of such operations with your legal team.
- **Dark Net:** Threat actors purchase compromised credentials from dedicated underground marketplaces or from sale offers through the Tor network. Some of these marketplaces require community vetting to access. Your organization can work with threat intelligence service providers to proactively identify credential data breaches which may impact your employees or customers.
- **Phishing campaigns:** Threat actors use large-scale phishing campaigns to harvest user credentials. These campaigns may mimic your corporate brand, and threat actors can use the harvested credentials to target your systems. You can subscribe to brand protection services to protect against campaigns targeting your corporate brand, and leverage threat intelligence services to track information on active phishing campaigns affecting your brand or industry sector.

- **Social engineering:** Threat actors use open-source research techniques to collect user account information and generate password lists. Public user profiles (e.g. social media) can reveal potential usernames, email addresses, or personal information that may provide hints on possible passwords.
- **System vulnerabilities:** Threat actors can exploit vulnerabilities present in web application systems to access credential information.

3 CREDENTIAL STUFFING PROTECTIONS

Defending against credential stuffing attacks requires a combination of measures, including risk-based and defense-in-depth approaches. MFA mechanisms, when implemented properly, will defeat credential stuffing attacks. However, threat actors can exploit additional vulnerabilities to bypass MFA protections. Below, we present some security control strategies that your organization should consider as recommended protections.

3.1 CORE SECURITY CONTROLS

In this section, we suggest core security controls that your organization can apply to reduce the risks associated with password authentication attacks. Depending on your organization's environment and security requirements, you may need to scope and tailor these controls or consider applying additional security controls. You can refer to Annex 3a of ITSG-33 [8] for the full catalogue of security controls.

3.1.1 MAINTAIN SOUND SECURITY POLICIES

Develop and implement a strong password policy.

Your password policy should mandate that your web application developers and users follow secure password principles. The policy should set minimum requirements for the composition and complexity of user passwords on your system. Review your policy regularly to provide updated guidance. Implement appropriate system tools to support enforcement activities and rectify policy violations. The policy should define security control requirements for the storage and management of user passwords. The list below highlights essential elements to consider in your password policy:

- Allow the use of all character types in passwords.
- Support MFA and mandate its use.
- Require the use of a unique password for each service.
- Ensure passwords meet attack-resistant complexity and composition requirements.
- Encourage the use of passphrases because the length of the password is likely more important than its complexity.
- Prevent the use of common passwords on your platform.
  - Keep an offline dictionary with a significant number of common passwords and conduct a lookup for each user created password, rejecting them if they match.
- Check for easy-to-guess, leaked or compromised passwords regularly and notify affected users.
- Force a password change if a leak or malicious activity is detected.
- Discourage the use of email address information as usernames on your platform.
- Notify account holders of suspicious login or security events, as well as the date and time when they last logged on.
- Activate account lockout or timeout policies to prevent automated login attempts.

  - The lockout policy should be specifically for the web application, not the active directory account, as it could allow a threat actor to lockout legitimate accounts.
- Deactivate unused or guest accounts and rename the default administrator account to something less easily guessed.
- Encourage the use of reputable password managers to safeguard passwords and facilitate the culture of using complex passwords.
- Review password policies to ensure they align with the latest GC, Cyber Centre, and OWASP guidelines.

Enable MFA mechanisms.

MFA is one of the most effective protections against password compromise attacks. In addition to using a password, your users must provide additional authentication factors (e.g. something they have or something they are). Some examples of a second factor include hardware tokens, public key infrastructure (PKI) certificates, digital security keys, or biometrics like fingerprints.

MFA adds protection in scenarios where a threat actor knows a valid username and password combination. However, MFA can be defeated through targeted attacks. An example is when your user is a victim of a phishing attack and provides all the authentication details required for the threat actor to gain access. To protect against this attack scenario, system owners can implement risk behaviour profiling mechanisms to detect malicious login requests using valid credentials.

Allow your users to create and modify their account names.

The use of email addresses as usernames is prevalent on many platforms. Threat actors can easily map publicly available credential breach data, using these email addresses with potential passwords. Using unique usernames on web platforms provides some additional protections against credential stuffing attacks. Refer to the following best practice when generating user identities for your website:

- Allow users to create unique user account names on your platform.
- Allow users to change their user account names if required.
- Allow users to use special characters in usernames.
- Generate unique user identifiers for your users if that is a preferred option.

Educate your users and implement tools to prevent insecure password practices.

Educate your users to create complex passwords and avoid reusing their passwords on other platforms. Bad password practices, such as password reuse, increase the likelihood of credential stuffing attacks. Design tools within your system to detect and prevent password reuse.

Although your application may keep records of old password hashes for validation, note that this may increase your compliance and data protection requirements. Implement appropriate safeguards to secure this data.

For more information on best practices for generating secure passwords, refer to the following:

- *ITSAP.30.032 Best Practices for Passphrases and Passwords* [10].

3.1.2 HARDEN YOUR AUTHENTICATION SYSTEMS

Harden authentication workflows and disable outdated algorithms.

Secure your application's authentication pathways. Identify logic errors in the authentication workflow that threat actors may exploit. A common issue is leaking account status information in feedback messages. Ensure that failed authentication messages do not reveal whether a user account exists or not. The execution response times should be consistent and should not leak information back to the user.

Strengthening your authentication workflows and removing outdated algorithms requires that you use tracking application packages in your system. You should remove all vulnerable or obsolete packages. Do not allow users to bypass any step in a multi-step authentication process. Implement proper session management controls.

Implement web application protections against known or unknown vulnerability exploits.

Protecting your web application against known and unknown exploits is crucial. Threat actors can launch targeted exploits, using vulnerabilities present within your system, to access your application's credential store. Solutions such as web application firewalls (WAFs) and web application proxies (WAPs) can mitigate the impacts of such attacks. To protect against injection attacks, implement user input validation techniques such as the use of parameterized queries and stored procedures. You can also consider enforcing Hypertext Transfer Protocol (HTTP) Strict Transport Security (HSTS) policies to force web browsers to access your web application using Hypertext Transfer Protocol Secure (HTTPS), instead of using plaintext HTTP. Enforcing HSTS protects against session intercept attacks.

Implement secure credential storage and management practices.

Fortifying the protections around password vaults keep related data breaches down. While data breaches may occur, your organization should architect its web application systems with protections to keep threat actors from easily accessing user credentials. Recommendations on how to securely manage your credential vault include the following:

- Store only encrypted or hashed copies of passwords; do not store password data in plaintext.
- Use cryptographic salt and pepper to secure and increase the complexity of passwords.
- Use hardware security modules (HSMs) or secret vaults to safely store encryption keys.
- Review applications regularly to remove the use of weak encryption algorithms.

You should review OWASP's *Application Security Verification Standard 4.0* [4] for details on how to securely architect your web application's authentication infrastructure.

Deploy rate-limiting controls such as the completely automated public Turing test to tell computers and humans apart (CAPTCHA).

Dynamic rate-limiting and throttling mechanisms are effective against large-scale, automated authentication attacks. You can use security mechanisms, such as CAPTCHAs, dynamic rate limits, and timeout triggers, to reduce or block scripted attacks. Historically, CAPTCHA controls are known to have adverse effects on user experience but recent advancements in the technology have led to significant improvements. Integration of behavioural analysis features and performing the CAPTCHA test in the background with no human interaction are some examples. Most implementation suggests activating

CAPTCHAs based on observed behaviours (e.g. suspicious logins, activity rate spikes, or known bot patterns). Sophisticated attacks may bypass CAPTCHA controls.

Consider account lockout policies to deter aggressive attacks. However, lockouts may inadvertently lead to distributed denial-of-service (DDoS) attacks on legitimate users. You may consider time-based or risk-based account lockout policies to reduce the impact on legitimate users.

3.1.3 BLOCK KNOWN ATTACK INDICATORS

Block requests from known malicious sources.

Block service requests from previously identified malicious sources. Based on your use case, consider permanently blocking Internet Protocol (IP) addresses known for malicious activity. Deploy automated bot management solutions to block traffic that matches known bot signatures.

Because threat actors change infrastructure often, using threat intelligence feeds can enhance your controls. Use IP reputation or threat intelligence services to gain insights about visitors to your application. For some applications, consider deny lists to block suspicious sources for a specified period rather than permanently. You may consider geofencing network controls for web applications with no use case outside a particular geographical region and implementing permanent blocks on traffic from regions known for malicious activity. Out of region access requests can be handled through an authorization process.

Implement credential breach monitoring to identify compromised credentials.

Use a credential breach monitoring solution as a proactive measure to identify compromised credentials in use on your platform. You can accomplish this by building an in-house solution or subscribing to a third-party service. There are risks associated with whichever model is selected. Maintaining additional sets of credentials for this purpose poses added compliance requirements and could make your system attractive to attackers. Third-party solutions typically involve sharing the password data (part or full), which raises privacy concerns. You should lock any accounts that are using compromised credentials and have the affected user re-verify their account and set a new password. Please refer to NIST SP 800-63B [2] for more information.

Track and block known malicious behaviour.

Threat actors use headless browsers and scripted tools to automate logins at scale. You can track these tools and map them to user activity to trigger dynamic-based block decisions. Your web application can block suspicious requests from malicious browser agents. Also, you can classify malicious behaviours detected on your platform to create future triggers and block matching activities.

Bot management solutions are options to evaluate when considering an integrated solution that covers both dynamic analysis and active threat protections. Bot management is a strategy that enables you to filter which bots are allowed to access your web assets. Bot management uses a range of security, machine learning, and web development technologies to accurately assess bots and block malicious activity while allowing legitimate bots to operate.

Use anomaly detection technologies to block suspicious activity.

Due to the sophistication and scale supported by many web applications, using anomaly detection solutions can help you identify and block suspicious logins. Assign risk scores to user requests by using user behaviour analytic solutions to capture mouse movements, screen swipes and typing patterns. Risk score outcomes that fall outside expected behaviours will trigger appropriate security control action. A block action is triggered when a malicious activity exceeds the score threshold. You can use statistical or frequency analysis to flag and prevent the use of weak passwords on your platform.

3.1.4 USE MODERN AUTHENTICATION SERVICES

Use online authentication federated services.

Using web-based authentication federated services such as Sign In Canada, GC Key, OpenID Connect and similar authentication-as-a-service platforms can offer alternative options to safely authenticate your users. Federation services allow your application to use a third-party's authentication service to identify users on your platform. This can reduce the cost, complexities, and limitations associated with managing an internal authentication solution. There are several digital identity projects in progress in the GC and Canada's provincial governments to encourage the use and acceptance of federation and trusted digital identities.

Federated identity services raise potential security risks. For example, a security event affecting the service provider's platform may have a cascading effect on your web application, making your web application inaccessible to users. Additionally, federating services may expose their users to privacy and confidentiality risks. Most implementations offer these external services as an authentication option and have fallback mechanisms to their internally supported authentication infrastructure.

Consider passwordless solutions.

Passwordless solutions use cases are growing. Web applications may allow their users to login using passwordless authentication solutions that use a combination of physical security keys, biometric data, cryptographic certificates, and physical device characteristics. Like most recommendations discussed, passwordless solutions are not applicable for every use case. This solution may be more appropriate for web applications accessed from a trusted device or network. In other use cases, this solution can be triggered to validate a user's request if a suspicious activity is detected. W3C Web Authentication (WebAuthn) is an example of a standard that supports the use of passwordless solutions in web applications. For more details on the WebAuthn specification, refer to *Web Authentication: An API for accessing Public Key Credentials Level 1* [11].

When deciding on authentication solutions, consider that some solutions might be associated with greater privacy concerns (e.g. biometrics) and require additional measures to ensure that privacy is adequately protected.

3.1.5 CONTINUOUSLY REVIEW YOUR APPLICATION'S TRUST RELATIONSHIPS

Implement continuous authentication and activity risk scoring.

Architect your application with assumptions that user credential compromises will occur. Design your web application to continuously assess user actions on the system and trigger re-authentication for high-risk events. A threat actor may use a successful phishing attack to obtain the password and the multi-factor code for the initial access. If your application triggers re-authentication requests on in-application actions, you can minimize the impact of this attack.

Also, you can append a risk score to every user action. If the overall risk score is below an expected threshold, re-authentication requests are triggered. Scenarios that can be evaluated to trigger re-authentication include first-time device use, device health changes, user profile changes, user role and rights changes, and administrative operations.

Use device attributes to authenticate and authorize actions.

You can use device attributes as additional data to secure the login process and authorize user requests. Network attributes such as geolocation data can also be useful. However, network attributes can be easily spoofed or circumvented using web proxies or virtual private networks (VPNs). Identifiers on access devices can be used to validate authentication and determine which services are available to the user. Additional security validation workflows can be triggered when a new device is detected. Device or source fingerprinting can be achieved via several measures including the following examples:

- Using physical device and configuration features (e.g. screen size, language settings, time zone settings).
- Using web browser user agent string or installed plug-ins information.
- Using network device information such as the media access control (MAC) address or Bluetooth device identifier information.
- Using Global Positioning System (GPS) data from the local device or using spoof resistant geolocation intelligence services.
- Creating a persistent cookie on the device for future identification.
- Using a third-party service to validate device health.

Overall, information returned from client-side fingerprinting cannot be trusted. Users may deploy tools to prevent information gathering or spoofing data to mislead the web application. You should weigh and deploy client-side fingerprinting results alongside other controls.

3.2 ADDITIONAL CONSIDERATIONS

3.2.1 LEGAL ENVIRONMENT

Align selected solutions to the legal environment in your jurisdiction of operation.

In some geographical jurisdictions, solutions proposed may inadvertently encounter legal constraints. You should evaluate the legal implications of the solutions you select. Your organization should understand how the controls its considering may impact users, legal requirements, and privacy risks.

GC departments and agencies should refer to the *Privacy Act* [12] and complete a Privacy Impact Assessment before implementing any solutions or changes. If your organization is not federally regulated, refer to the *Personal Information Protection and Electronic Documents Act* [13], which is the regulatory framework that sets out the ground rules for how private organizations should handle personal information, or the appropriate provincial privacy legislation, should one apply.

In some circumstances, your operations may require compliance with regulations in other geographical jurisdictions. If your web service receives data from European citizens, you are required to comply with the *General Data Protection Regulation* [14]. Consult with your legal team and ensure you fully understand the implications of your operational footprint.

4 CONCLUSION

Implementing MFA will protect against a vast majority of credential stuffing attacks. Traditional controls such as malicious network filtering and blocking malicious browser agents are largely becoming ineffective against new techniques. Modern authentication methods will provide alternative options for consideration. To protect your organization from credential stuffing attacks, consider the following techniques:

- Harden your authentication workflows;
- Disabling outdated algorithms;
- Block requests from known malicious sources;
- Implement credential breach monitoring to identify compromised credentials; and
- Use anomaly detection technologies to block suspicious activity.

Your web application's business need and architecture will be crucial in determining which solutions are selected. A multi-faceted approach will provide the best protection.

5 SUPPORTING CONTENT

5.1 LIST OF ABBREVIATIONS

Term	Definition
API	Application programming interface
CAPTCHA	Completely automated public Turing test to tell computers and humans apart
DDoS	Distributed Denial of Service
GC	Government of Canada
GPS	Global Positioning System
HSM	Hardware security module
HSTS	Hypertext Transfer Protocol Strict Transport Security
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IoT	Internet of things
IP	Internet Protocol
IT	Information Technology
MAC	Media access control
MFA	Multi-factor authentication
NIST	National Institute of Standards and Technology
OWASP	Open Web Application Security Project
PKI	Public key infrastructure
SIM	Subscriber identity module
SMS	Short message service
SP	Special Publication (NIST)
SS7	Signaling System 7
SSO	Single sign-on
VPN	Virtual private network
WAF	Web application firewall
WAP	Web application proxies
WebAuthn	W3C Web Authentication

5.2 Glossary

Term	Definition
Application programming interfaces	A system access point or library function that has a well-defined syntax and is accessible from application programs or user code to provide well-defined functionality.
Authentication	A process or measure used to verify a user's identity.
Dynamic rate limits	Dynamic controls used to control the rate of requests sent or received by a web server.
Federated identity	A process that allows for the conveyance of identity and authentication information among systems.
HTTP Strict Transport Security (HSTS)	A website access policy that informs user agents that the website should only be accessed over the HTTPS protocol. The policy helps protect websites against eavesdropping attacks.
Identity proofing	The process of collecting, validating, and verifying a user's identity information to establish credentials for accessing a system.
Multi-factor authentication (MFA)	A process for verifying a user's identity by using different mechanisms, including something the user knows, something the user has, or something the user is.
Phishing	An attempt by a third party to solicit confidential information from an individual, group, or organization by mimicking or spoofing a specific, usually well-known brand, usually for financial gain. Phishers attempt to trick users into disclosing personal data, such as credit card numbers, online banking credentials, and other sensitive information, which they may then use to commit fraudulent acts.
Plaintext	Unencrypted information.
Public Key Infrastructure	The architecture, organization, techniques, practices, and procedures that collectively support the operation of a certificate-based public key cryptographic system.
Signaling System 7 (SS7)	An international signalling system that is used to manage data transfers on cellular devices.
Virtual private network (VPN)	A private communications network usually used within a company, or by several different companies or organizations, to communicate over a wider network. VPN communications are typically encrypted or encoded to protect the traffic from other users on the public network carrying the VPN.
Vulnerability	A flaw or weakness in the design or implementation of an information system or its environment that could be exploited to adversely affect an organization's assets or operations.

5.3 REFERENCES

Number	Reference
1	Canadian Centre for Cyber Security. *ITSP.30.031 v3 User Authentication Guidance for Information Technology Systems*.
2	National Institute of Standards and Technology. *NIST SP 800-63B Digital Identity Guidelines: Authentication and Lifecycle Management*. June 2017.
3	National Institute of Standards and Technology. *NIST SP 800-95 Guide to Secure Web Services*. August 2007.
4	Open Web Application Security Project. *OWASP Application Security Verification Standard 4.0.2*. October 2020.
5	National Institute of Standards and Technology. *NIST SP 800-63-3 Digital Identity Guidelines*. June 2017.
6	National Institute of Standards and Technology. *NIST SP 800-63A Digital Identity Guidelines: Enrollment and Identity Proofing*. June 2017.
7	National Institute of Standards and Technology. *NIST SP 800-63C Digital Identity Guidelines: Federation and Assertions*. June 2017.
8	Canadian Centre for Cyber Security. *ITSG-33 IT Security Risk Management: A Lifecycle Approach.* November 2012.
9	Janca, Tanya. *Alice and Bob Learn Application Security*. November 2020. (Print resource).
10	Canadian Centre for Cyber Security. *ITSAP.30.032 Best Practices for Passphrases and Passwords.* September 2019
11	W3C. *Web Authentication: An API for Accessing Public Key Credentials Level 1*. March 2019.
12	Canada. Department of Justice. *Privacy Act*.
13	Canada. Department of Justice. *Personal Information Protection and Electronic Documents Act*.
14	European Union. *General Data Protection Regulation*.