

TIBCO WebFOCUS®

Security and Administration Best Practices

Release 9.0.0 and higher

October 2022

DN4501763.1022



Contents

| | |
|--|-----------|
| 1. Security Best Practices Overview | 5 |
| Overview | 5 |
| Additional Resources | 10 |
| 2. Secure the TIBCO WebFOCUS Infrastructure | 11 |
| Overview | 11 |
| Best Practice | 14 |
| Additional Resources | 14 |
| 3. Configure TIBCO WebFOCUS Reporting Server Settings | 15 |
| Overview | 15 |
| Best Practice | 18 |
| Additional Resources | 19 |
| 4. Configure TIBCO WebFOCUS Client Settings | 21 |
| Overview | 21 |
| Best Practice | 24 |
| Additional Resources | 24 |
| 5. Secure Communications Between Browsers and the TIBCO WebFOCUS Client | |
| Using HTTPS and HSTS | 25 |
| Overview | 25 |
| Best Practice | 26 |
| Additional Resources | 26 |
| 6. Implement Secure Authentication and User Account Policies | 27 |
| Overview | 27 |
| Best Practice | 28 |
| Additional Resources | 28 |
| 7. Use TIBCO WebFOCUS Resource Templates and TIBCO WebFOCUS Reporting | |
| Server Access Control Templates | 29 |
| Overview | 29 |
| Best Practice | 30 |
| Additional Resources | 30 |
| 8. Build Secure Applications | 31 |

- Overview 31
- Best Practice 42
- Additional Resources 42
- 9. Secure Apache Solr Communications43**
 - Overview 43
 - Best Practice 44
 - Additional Resources 45
- A. Glossary47**
- Legal and Third-Party Notices55**

Security Best Practices Overview

This overview provides a summary of the best practices you should follow as you secure an installation of TIBCO WebFOCUS® and administer ongoing application development. The recommendations listed in this summary can be applied by WebFOCUS® administrators and WebFOCUS application developers.

In this chapter:

- ❑ [Overview](#)
 - ❑ [Additional Resources](#)
-

Overview

As defined in the National Information Assurance Glossary, published by the Committee on National Security Systems, *Information Assurance* refers to measures that protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation. You can access this and other terms related to security at the following link: https://www.dni.gov/files/NCSC/documents/nittf/CNSSI-4009_National_Information_Assurance.pdf

To that end, a comprehensive Software Development Lifecycle (SDLC) that focuses on eliminating security vulnerabilities and improving the security posture of the WebFOCUS tier of products was implemented. Any vulnerabilities found after each round of testing are prioritized based on their severity and risk level and fixed prior to product release. The SDLC process includes, but is not limited to:

❑ Static Application Security Testing (SAST)

Static Application Security Testing evaluates application source code vulnerabilities. Using third-party proprietary and open-source SAST tools, product source code is scanned during the build process to ensure that core product features are free from security vulnerabilities.

❑ Code Review

Code returns are reviewed and authorized by engineering team leads and management to ensure that they adhere to secure coding practices, and that they introduce no new bugs or security vulnerabilities into the product.

❑ **Open Web Application Security Project (OWASP) Dependency-Check**

All third-party libraries are reviewed against the National Vulnerability Database, as well as enterprise class software that provides the same type of functionality, to identify any vulnerabilities.

❑ **Dynamic Application Security Testing (DAST)**

Dynamic Application Security Testing evaluates the operational vulnerabilities of product applications. Third-party DAST tools run against live sample WebFOCUS environments to ensure that they are free from security vulnerabilities.

❑ **Interactive Application Security Testing (IAST)**

Third-party IAST tools are used to instrument live WebFOCUS environments, review Java memory and performance metrics, and ensure that no Denial of Service issues are present.

❑ **Ethical hacking techniques**

Engineering and Quality Assurance teams attempt to break application security by applying ethical hacking techniques that expose such security issues as XSS, XML Entity, SQL, LDAP, OS injection attack vulnerabilities or problems with authentication or authorization.

❑ **Independent Security Audits**

Security assessors from the Security Engineering team, an independent Internal Testing Team of TIBCO, highly trained in web application development and security issues, audit live WebFOCUS environments to confirm that they adhere to secure processes. The assessors also conduct penetration tests to confirm that the product is free from security vulnerabilities.

❑ **Annual Staff Security Training**

All staff members must attend annual security training courses that cover Phishing Fundamentals, CEO Fraud attacks, the European Union's General Data Protection Regulation (GDPR), and other current environmental risks. In addition, Engineering staff attend classes that focus on secure coding practices.

❑ **Prioritization of Security Bugs**

Whenever security issues are identified after a product release, the Customer Support team works closely with customers to triage these issues by severity and risk, and works closely with Product Management, Engineering, and Quality Assurance to ensure that they are remediated as quickly as possible via the standard Inverse Reinforcement Learning (IRL) process.

In addition, unit tests are used within the development process and during the introduction of new features or bug fixes to ensure that newly introduced code is fully functional, and integration tests of multiple components are conducted by Quality Assurance (QA) Automation teams.

The use of Agile best practices and Continuous Integration/Continuous Development (CI/CD) processes ensures that the WebFOCUS product can be released on a two-week cadence to respond to critical customer bugs and deliver a limited set of new features. CI/CD practices provide the following advantages.

☐ **Code Development**

Based on product management feature specifications, the engineering team can implement code development with functionality, scalability, performance, and security in mind. Code submitted by team members is reviewed by engineering management, and all pull requests are logged and authorized.

☐ **Product Build**

WebFOCUS and related products are built in conjunction with all necessary third-party libraries and APIs, which can be installed to support live QA testing.

☐ **Unit Tests**

Static Application Security Testing using various code scanning tools is included in the unit testing stage of all features and previously-fixed bugs. Any security failures identified at this point are addressed by the Engineering team and returned to the lead development track, to be built again. Daily Security Alerts are reviewed on industry web sites to look for announcements of potential threats to our products and customers.

☐ **Deployment to Staging Environment/Formal QA**

Staging environments are configured with different underlying platforms and versions (Windows, Linux, iSeries, zSeries, Containers, Application Servers, Cloud) and different authentication configurations (SAML, OIDC, Kerberos, IWA) to ensure that functional tests are applied successfully within a variety of deployments.

☐ **Functional Tests**

Dynamic Application Security Testing, Interactive Application Security Testing, and ethical hacking techniques are performed along with functional testing of implemented features. The engineering team fixes any failures identified at this point, and returns the revised code to the lead development track for reassembly. All third-party licenses included in the product are also reviewed to ensure that there are no license violations.

☐ **Deployment to Production Environment/Software Release**

Once all development and testing is done, and all features are deemed working, the product is posted for use in on-premise applications, or deployed to our Cloud Infrastructure. The time required to complete the workflow from development to product release is two weeks.

Besides the implementation of this SDLC, WebFOCUS features a number of built-in security capabilities that emphasize risk management and defend against malicious hacker attacks, which are critical to external web-based Business Intelligence applications. These capabilities include, but are not limited to the following features.

☐ CSRF Protection

☐ Validation of regular expressions and database input operations

☐ Encoding of HTML output data

☐ Protection against injection flaws, including XML Entity Injection, Null Byte Injection, and Log injection

Just as a security SDLC process is implemented when manufacturing the WebFOCUS product, secure WebFOCUS applications need to incorporate similar methodology. When developing applications with WebFOCUS, security cannot be an afterthought. Instead, a series of installation, implementation, and development best practices need to be followed, to ensure that the applications developed are secure and free from security vulnerabilities. To support this effort, this document identifies best practices for developing and maintaining secure WebFOCUS applications.

These best practices are not absolute. Depending on the type of data being accessed, the user community, and if the product is deployed as an external or internal application, some or all of these best practices may be needed.

Review the following security recommendations and individual best practices that are outlined in subsequent topics for more information.

☐ Secure the WebFOCUS infrastructure.

☐ Establish WebFOCUS components on separate machines.

☐ Limit the services or processes available on the computer that supports each WebFOCUS component.

☐ Ensure that the latest versions and patches of all software used within the environment are installed.

- ☐ Change the default credentials on all components.
- ☐ Establish the use of HTTP Secured (https) technology on communications from the browser to the TIBCO WebFOCUS® Client.
- ☐ Configure all relevant settings on the WebFOCUS® Client and TIBCO WebFOCUS® Reporting Server that are feasible within your installation of WebFOCUS.
- ☐ Establish Web Application Firewalls and Firewalls between all components of the network.
- ☐ Secure WebFOCUS content as required by the level of security imposed upon your environment.
- ☐ Configure WebFOCUS® Reporting Server settings:
 - ☐ Ensure that the execution of operating system commands is disabled.
 - ☐ Ensure that the execution of direct SQL Passthru commands is disabled.
 - ☐ Ensure that the UINFO privilege is selected to remove detailed error messages issued to end users.
 - ☐ Encode HTML output data by setting the HTMLencode setting to ON.
 - ☐ Disable echo output by setting the DEFecho setting to NONE.
 - ☐ Establish IP Restriction filtering by setting the RESTRICT_TO_IP keyword in the Reporting Server communications configuration file (odin.cfg) to accommodate TCP/IP and HTTP access.
 - ☐ Establish Dynamic DBA security for the repository database.
 - ☐ Encrypt data at rest.
- ☐ Configure WebFOCUS Client settings:
 - ☐ Ensure that the HttpOnly flag is set to true for JSESSIONID.
 - ☐ Disable public access in the Default Security Zone.
 - ☐ Encrypt communications between the Client and the Reporting Server.
 - ☐ Set the session timeout to the smallest possible value, based on the context of the application, and activate the automatic sign-out feature.
 - ☐ Remove detailed error messages issued to end users.

- ☐ Secure communications between browsers and the Client using HTTPS.
- ☐ Implement secure authentication and user account policies.
- ☐ Use WebFOCUS Resource Templates and Reporting Server Access Control Templates.
- ☐ Build secure applications:
 - ☐ Protect against injection flaws and cross-site scripting attacks.
 - ☐ Add the CHKfmt function to all applications that include character string database input operations.
 - ☐ Add the CHKnum function to all applications that include numeric string database input operations.
 - ☐ Add the GETtok function to all applications that include database input operations that extract character or numeric strings from larger data for validation.
 - ☐ Add regular expression validation to all database input operations.
 - ☐ Additionally, for cross-site scripting, add the XMLencod function to all numeric and character string database input operations.
 - ☐ Configure the application to neutralize HTML tags within untrusted data in the database.
 - ☐ Protect against cross-site request forgery.
 - ☐ Protect against click-jacking attacks.
 - ☐ Protect against sensitive information exposure.
 - ☐ Incorporate security protections into the application development process.

Additional Resources

For more information about configuration and implementation topics, see the *TIBCO WebFOCUS® Security and Administration* technical content, and the *TIBCO WebFOCUS® Reporting Server Administration* technical content. For more information about application development topics, see the *TIBCO WebFOCUS® Creating Reports With TIBCO WebFOCUS® Language* technical content, the *TIBCO WebFOCUS® Developing Reporting Applications* technical content, the *TIBCO WebFOCUS® Describing Data With TIBCO WebFOCUS® Language* technical content, and the *TIBCO WebFOCUS® Using Functions* technical content.

For more information about Open Web Application Security Project (OWASP), see <https://owasp.org/www-project-top-ten/>

Secure the TIBCO WebFOCUS Infrastructure

This best practice addresses the way in which administrators can secure the environment when deploying WebFOCUS for operations and application development. It can be applied by a WebFOCUS administrator.

In this chapter:

- ☐ [Overview](#)
 - ☐ [Best Practice](#)
 - ☐ [Additional Resources](#)
-

Overview

The architecture of the WebFOCUS network creates a secure environment for WebFOCUS operations and for application development. Security settings configured on the Reporting Server, Application Server, and Client also contribute to secure application development. Administrators need to incorporate these protections into their environment to ensure the maximum level of security.

The Java Spring Framework platform and Session Management controls are fundamental components of the WebFOCUS security architecture. The Spring Framework platform provides comprehensive infrastructure support for Java applications, and accommodates a wide variety of security features, including multiple pre-authentication methods and session management capabilities. Security features within the WebFOCUS product include support for both authentication and authorization, protection against attacks like session fixation, directory traversal, cookie replay, information disclosure, click-jacking, cross-site request forgery, cross-site scripting, and injection flaws.

We recommend that you incorporate the following practices into your deployment of WebFOCUS. Even though all of these practices are generally recommended, administrators may not be able to implement one or more of them due to limitations on resources or the relevance to their specific configuration.

1. Establish a secure infrastructure.

- a. Establish WebFOCUS components on separate machines.

Install the WebFOCUS Client, the Reporting Server, the Repository RDBMS, as well as the Operational RDBMS, on separate machines.

This separation helps ensure that if any single component is compromised, the remaining components will be protected.

- b. Limit the services or processes available on the computer that supports each component.

This means that no service or process that does not support WebFOCUS operations should be enabled.

- c. Ensure that the latest versions of all operating systems and software applications are installed.

By incorporating the latest versions, you help ensure that the operating system and all supporting applications on each machine are protected against the latest security threats.

- d. Change the default credentials on all WebFOCUS components.

Replace the default credentials of all operating systems, databases, and other applications with credentials known only to parties in your organization. This update ensures that all components and supporting applications are protected from intrusion by unauthorized parties using these default credentials.

- e. Establish the use of HTTP Secured (https) technology on communications from the browser to the Client.

Https is highly recommended to ensure the privacy, authenticity, and data integrity of the communications between browsers of end users and the WebFOCUS Server, especially for applications, such as those that require users to supply credentials, that require a higher level of security and authentication than that required by public access.

- f. Configure all relevant settings on the WebFOCUS Client and Reporting Server that are feasible within your installation of WebFOCUS.
- g. Establish Web Application Firewalls and Firewalls between all components of the network.

2. Secure WebFOCUS content as required by the level of security imposed upon your environment.

a. Totally Secured

Production Environment: Permits no development and no new My Content. For this environment:

- ☐ Basic users run externally developed applications and can create ad hoc versions of reports or other features using guided tools, such as the Autoprompt interface.
- ☐ Ensure that the roles assigned to the basic user and advanced user groups include the following limitations:
 - ☐ Prevent the development of new Standard Reports or Reporting Objects.
 - ☐ Prevent the development of new Private Reports in the (My Reports) folder.
 - ☐ Include the No Save Selection requirement.
 - ☐ Prevent the development of new reports, charts, or other content.
- ☐ Limit the level of detailed system information included in messages by assigning the value of Expected or None to the Message Detail (IBI_MESSAGE_DETAIL) setting. Disable the Reporting Server UINFO privilege to minimize basic and advanced user access to the server.

Development Environment: Permits strictly controlled development. For this environment:

- ☐ Limit the development of standard reports to a separate environment dedicated to development.
- ☐ Conduct functionality tests in the Test environment and not in the Development environment.
- ☐ Conduct penetration and vulnerability tests in the Test environment.
- ☐ Use the Change Management feature to transfer all applications from the Development to the Production environment.
- ☐ Follow a strictly controlled process for all application development.

b. **BI Secured:**

Production Environment: Advanced types of users can create their own content in the My Content folder, such as ad hoc reports. For this environment:

- ☐ Prevent the development of new Standard Reports or Reporting Objects.
- ☐ Limit the development of new Private Reports to those in the (My Reports) folder or in InfoAssist.
- ☐ Limit Sharing to developers working on the same content.
- ☐ Do not assign developers to the production environment.
- ☐ Control additions to standard report content.
- ☐ Develop Private Content using BI Tools.

Development Environment: Only Developers can create standard reports and reporting objects. For this environment:

- ☐ Limit the development of standard reports to a separate, dedicated Development environment.
- ☐ Conduct functionality and vulnerability tests in the Test environment.
- ☐ Use the Change Management feature to transfer all applications from the Development to the Production environment.
- ☐ Follow a strictly controlled process for all application development.

Best Practice

We recommend that all users establish a secure infrastructure, secure WebFOCUS content, and incorporate security protections into the application development process, as described in this document, into their WebFOCUS deployments.

Additional Resources

For more information, see the *TIBCO WebFOCUS® Security and Administration* technical content and the *TIBCO WebFOCUS® Developing Reporting Applications* technical content.



Chapter 3

Configure TIBCO WebFOCUS Reporting Server Settings

This best practice addresses how to configure WebFOCUS Reporting Server Settings to improve security based on the requirements of your application. It can be applied by a WebFOCUS Administrator.

In this chapter:

- ☐ [Overview](#)
 - ☐ [Best Practice](#)
 - ☐ [Additional Resources](#)
-

Overview

The WebFOCUS Reporting Server is installed with several features that are designed to protect against the most common and dangerous forms of security attack. In order to take full advantage of these settings, we recommend that you adopt the following configurations that best support the requirements of your application.

1. **Ensure that the execution of operating system commands is disabled.**

Disabling this capability protects the operating system of the Reporting Server by preventing WebFOCUS users from executing operating system commands that would write to the Reporting Server. However, before disabling this capability, consult with your application development team to ensure that the applications they are developing do not require it.

The Reporting Server defines four basic roles on the Access Control page: Server Administrator, Server Operator, Application Administrator, and Basic User. Individual installations of WebFOCUS may also define customized roles that use the WebFOCUS application. For each basic and customized role, administrators must ensure that the NOSYS (Disable Operating System Commands) setting is selected.

Do not include the Server Administrator role in this review, however. The Server Administrator role is fully authorized to execute operating system commands.

2. Ensure that the execution of direct SQL Passthru commands is disabled.

Disabling this capability protects the database of the Reporting Server by preventing WebFOCUS users from executing SQL Passthru commands that would affect it directly. However, before disabling this capability, consult with your application development team to ensure that the applications they are developing do not require it.

If your application requires the use of SQL commands, the use of SQL Stored procedures is preferable to Direct SQL Passthru commands and is highly recommended. If it is not possible to use stored procedures, ensure that all SQL Passthru commands included in application code contain appropriate security protections, and that they accept parameters securely.

The Reporting Server defines four basic roles on the Access Control page: Server Administrator, Server Operator, Application Administrator, and Basic User. Individual installations of WebFOCUS may also define customized roles that use the WebFOCUS application. For each basic and customized role, administrators must ensure that the NODPT (Disable Direct Passthru) setting is selected.

Do not include the Server Administrator role in this review, however. The Server Administrator role is fully authorized to execute direct SQL Passthru commands.

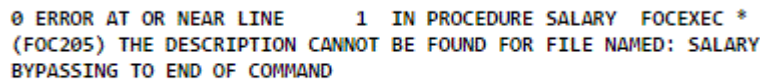
3. Ensure that the UINFO privilege is selected to remove detailed error messages issued to end users.

Activating the UINFO privilege helps protect against an information exposure vulnerability that results from the display of fully-detailed error messages to basic users.

For the Reporting Server basic user role, and any customized roles that are based on it, administrators must select the UINFO (Disable Display My Console, Error Message, Logon Info, Server Version, Console Log and Help) setting. Do not include the Application Administrator, Server Operator, or Server Administrator role in this update, however.

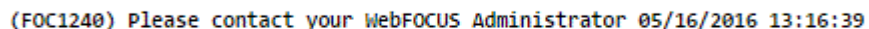
This best practice ensures that the Reporting Server automatically limits the display of detailed error messages to any user who accesses the Reporting Server through the basic user role. This update makes the description of specific server errors more generic so that runtime errors generate generic messages for end users. For example:

- ☐ If UINFO is not selected, WebFOCUS returns the following detailed error message, as shown in the following image.



```
0 ERROR AT OR NEAR LINE      1 IN PROCEDURE SALARY FOCEXEC *
(FOC205) THE DESCRIPTION CANNOT BE FOUND FOR FILE NAMED: SALARY
BYPASSING TO END OF COMMAND
```

- ☐ If UINFO is selected, WebFOCUS returns the following generic error message, as shown in the following image.



```
(FOC1240) Please contact your WebFOCUS Administrator 05/16/2016 13:16:39
```

4. **Encode HTML output data by setting the HTMLENCODE setting to ON.**

Encoding HTML tags included within data as plain text prevents them from being interpreted as executable code by the browser that retrieves data from the Reporting Server. This best practice prevents an attack on the server resulting from the insertion of executable code into data.

We recommend that you activate the HTMLENCODE setting by assigning it a value of On. Doing so disables the rendering of HTML tags within a browser when these tags are stored within the actual data, or when they are created using a DEFINE or COMPUTE command.

5. **Disable echo output by setting the DEFECHO setting to NONE.**

When you disable echo output, you help prevent information regarding WebFOCUS code from being returned to the user.

When activated, the DEFECHO setting causes the Reporting Server to display procedure code while an application is running. Even though this capability is helpful during application development and troubleshooting, once development and troubleshooting are complete, it needlessly exposes procedure code to malicious parties.

We recommend that you set the DEFECHO Setting to None on the Reporting Server to prevent the delivery of echo output from the WebFOCUS applications it hosts to the browsers of end users, by default. After the value of DEFECHO has been set to NONE, it cannot be changed during a session or connection.

6. Establish IP Restriction filtering by setting the RESTRICT_TO_IP keyword in the Reporting Server communications configuration file (odin.cfg) to accommodate TCP/IP and HTTP access.

By establishing IP Restriction filtering, you limit the range of incoming WebFOCUS Clients to clients who use TCP/IP protocols and whose addresses are known to you.

The Hypertext Transmission Protocol, (HTTP), Transmission Control Protocol (TCP) and Internet Protocol (IP) are all well-established data transmission methods that include built-in protections for data integrity and transmission reliability. Their use is so widespread that they are fundamental requirements for all applications. Attempts to communicate without using these protocols and the basic protections they provide risk exposing the application to the transmission of data from an unknown source and of a dubious quality.

We therefore recommend the restriction of all Reporting Server communications to the addresses of clients that are known to your organization.

7. Establish Dynamic DBA security for the repository database.

When you establish Dynamic DBA Security, you protect the repository database from incursion by unauthorized users. For example, you can limit access to a given data source by user name, and within that privilege, establish limits on the ability of a specified user to read, write, or update data. You can also restrict a user to certain fields or segments of data and ensure that only records that pass a validation test are retrieved.

8. Encrypt Data at Rest.

By setting the io_encryption setting to Y on the Reporting Server, you activate encryption for all binary, alpha, and delimited HOLD files in the edatemp and foccache directories. You also activate encryption for data agent trace files (tsxxx.trc) and agent output files (tsxxx.tro). After encryption, the extensions for these files are .trce and .troe, respectively. Encryption prevents users from opening these files in editors outside of the Web Console or Data Management Console and reading their contents. Trace files and agent output files can be viewed from the Web Console Workspace page under Traces And Logs.

Best Practice

We recommend that you incorporate those protections described in this document that are feasible and that do not compromise performance into your configuration of the WebFOCUS Reporting Server.

Additional Resources

For more information, see the *TIBCO WebFOCUS® Security and Administration* technical content.

Configure TIBCO WebFOCUS Client Settings

This best practice addresses how to configure additional WebFOCUS Client Settings to improve security based on the requirements of your application. It can be applied by a WebFOCUS administrator.

In this chapter:

- ☐ [Overview](#)
 - ☐ [Best Practice](#)
 - ☐ [Additional Resources](#)
-

Overview

The WebFOCUS Client is installed with several features that are designed to protect against the most common and dangerous forms of security attack. In order to take full advantage of these settings, we recommend that you adopt the following configurations that best support the requirements of your application.

1. Replace the default Password assigned to the Server Administrator with a secure password.

The password for the Server Administrator account is pre-configured during the WebFOCUS installation process to be the same as the password you supplied for the original administrator account. This default password is not secure, and it must be replaced by a password known only within your organization. You can replace the default password by typing a new password in the Password field on the External page of the Administration Console Security tab.

2. Change the default credentials on all WebFOCUS components.

Replace the default credentials of all operating systems, databases, and other applications with credentials known only to parties in your organization. This update ensures that all components and supporting applications are protected from intrusion by unauthorized parties using these default credentials.

3. Ensure that the HttpOnly flag is set to true for JSESSIONID.

This best practice ensures that all sessions will automatically include the HttpOnly flag in their JSESSION ID cookie.

For example:

```
Set-Cookie: JSESSIONID=BD61C838569C30474977ACDE3DAD8F54; Path=/ibi_apps/; HttpOnly.
```

The presence of this flag prevents malicious client side JavaScript from accessing the JSESSION ID cookie.

Note: If you use Apache Tomcat as your application server, this flag is set to true during the installation. If you use another application server, this flag may not be set to true. Administrators must therefore review and verify or update this setting in the deployment descriptor file.

4. **Disable public access in the Default Security zone.**

Disabling Public Access helps ensure that only authenticated users can gain access to WebFOCUS.

Public Access makes resources in the WFC/Repository/Public folder and procedures on the WebFOCUS Reporting Server available to all users. The Reporting Server credentials used for Public Access are specified in the Reporting Server Anonymous User ID (IBI_WFRS_Anonymous_User) setting and the Reporting Server Anonymous Password (IBI_WFRS_Anonymous_Pass) setting. However, even this limited amount of unauthenticated access could potentially enable unauthorized individuals to attempt to gain access to the WebFOCUS client.

Therefore, Public Access is disabled in all security zones except the Default Security Zone, by default. To fully protect the Client, we recommend that, unless there is a compelling reason to grant public access to WFC Resources or to procedures on the Reporting Server, administrators change this default configuration and disable Anonymous Access in the Default Security Zone.

5. **Encrypt communications between the WebFOCUS Client and the Reporting Server.**

Encrypted communications protect the integrity of information exchanged between the WebFOCUS client and the Reporting Server. Messages exchanged between the WebFOCUS Client and the Reporting Server can contain sensitive or confidential information, both in requests and responses. Malicious attacks against internal communications can attempt to extract or corrupt this information.

Therefore, we recommend that administrators configure the Encryption setting, located in the Server Connections page of the Administration Console Configuration tab, to accept, at a minimum, an Advanced Encryption Standard (AES) 128-bit encryption key. Administrators of installations that require more intensive encryption can select a higher-level encryption key.

6. Set the session timeout to the smallest possible value, based on the context of the application, and activate automatic sign-outs.

By reducing the time in which sessions remain open and idle, you reduce the time that they are vulnerable to intrusion from unauthorized parties, and minimize system exposure. By activating the automatic sign-out feature, you also ensure that users are redirected to a legitimate sign-out page identified in your WebFOCUS configuration when idle sessions time out.

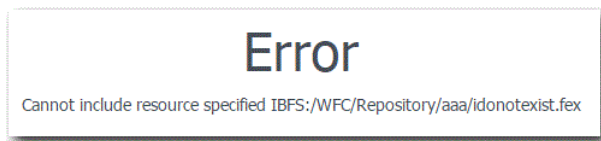
This Session Timeout setting is located on the BI Portal page of the Administration Console Configuration tab. As a best practice, we recommend that you replace the default value in that setting with the shortest period of time that will accommodate the length of most sessions. For secure applications, we generally recommend a maximum Session Timeout limit of 15 minutes.

The Enable Auto Sign-out setting and the Idle Timeout message duration (minutes) setting also appear on the BI Portals page of the Administration Console Configuration tab. As a best practice, we recommend that you select the check box in the Enable Auto Sign-out setting to activate the use of automatic sign-outs that direct users to a legitimate sign-out page when idle sessions time out. Users can then return to a new session from the sign-out page after being properly authenticated. We also recommend that you assign a value to the Idle Timeout message duration (minutes) setting that is less than or equal to the number of minutes in the Session Timeout setting to ensure that users receive an adequate advance warning of an impending timeout.

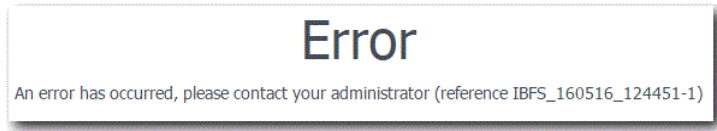
7. Remove detailed error messages issued to end users.

By removing detailed system information from WebFOCUS Client error messages, you prevent the potential disclosure of specific error message details, error descriptions, and other information that a malicious party could extract and use to launch additional attacks. Administrators can replace these detailed error messages with simplified error messages that only convey relevant information. To do so, they can assign a value of None to the Message Detail (IBI_MESSAGE_DETAIL) setting on the BI Portal page of the Administration Console Configuration tab.

When the value is set to Severe, the default setting, a fully-detailed error message displays, as shown in the following image.



When the value is set to None, details are excluded from the error message, as shown in the following image.



Best Practice

We recommend that you incorporate those protections described in this document that are feasible and that do not compromise performance into your configuration of the WebFOCUS Client.

Additional Resources

For more information, see the *TIBCO WebFOCUS® Security and Administration* technical content.

Secure Communications Between Browsers and the TIBCO WebFOCUS Client Using HTTPS and HSTS

This best practice addresses the need to secure connections between the browsers of WebFOCUS users and the WebFOCUS Client to ensure the confidentiality, authenticity, and integrity of all communications. It can be applied by a WebFOCUS administrator.

In this chapter:

- ☐ [Overview](#)
 - ☐ [Best Practice](#)
 - ☐ [Additional Resources](#)
-

Overview

Hypertext Transfer Protocol over Secure Socket Layer (https) technology, which includes the Transport Layer Security (TLS) protocol previously known as Secure Socket Layer (SSL), establishes a secure connection between users and the WebFOCUS client. It adds confidentiality, integrity, and authenticity to communications by:

- ☐ Encrypting any communication between browsers assigned to end users and the WebFOCUS Client.
- ☐ Ensuring that any data sent between browsers and the WebFOCUS Client is not tampered with or modified in any way.
- ☐ Validating that a user is communicating directly with the WebFOCUS Client and not with an impostor.

An HTTP Strict Transport Security (HSTS) policy is a security enhancement issued by a server that requires the use of the https protocol for all incoming requests. When this policy is in place, the server that hosts a website responds to the first request from a browser that does not use the https protocol by returning a message with a response header that contains the Strict-Transport-Security field. The presence of this field in the response header indicates that the server will not accept any further requests from that browser that do not arrive over an https connection.

The response header can also include a field identifying the time limit, typically one year, over which the policy will be enforced. Any subsequent requests from that browser that do not use this protocol will receive an error message in response.

When the browser receives a message with a response header that contains a Strict-Transport-Security field, it knows to use the https protocol when sending any future messages to the site. The browser also knows that any other site using the same name that does not require the use of this protocol is not legitimate, and it automatically redirects requests to the site that does require the use of the https protocol.

By imposing this policy within a WebFOCUS Security Zone, you introduce this extra level of security to all communications between users in that zone and the Application Server. The policy ensures that all communications within that zone use the https protocol and are therefore encrypted and validated by a public key certificate. It also helps prevent requests from users in that zone from being inadvertently misdirected to an illegitimate site that does not require the https protocol.

Best Practice

We recommend that you establish secure connections between all browsers assigned to end users, the WebFOCUS Client, and any other server that requires a secured connection. We also recommend that you establish the HTTP Strict Transport Security (HSTS) policy for all users.

The establishment of the SSL/TLS protocol is a detailed process that requires a secure implementation. If you are using Apache Tomcat as your web server, follow the procedures in the Configure WebFOCUS for SSL topics, which are located in the Security and Administration technical content. If you are using a web server from a different vendor, follow the instructions describing how to configure SSL/TLS for that server provided by your vendor.

Additional Resources

For more information, see the *TIBCO WebFOCUS® Security and Administration* technical content.

Implement Secure Authentication and User Account Policies

This best practice addresses the need to establish account policies that enhance secure user access to the WebFOCUS environment by establishing strong internal user account policies or by adopting third-party authentication providers. It can be applied by a WebFOCUS administrator.

In this chapter:

- ☐ [Overview](#)
 - ☐ [Best Practice](#)
 - ☐ [Additional Resources](#)
-

Overview

User accounts and passwords are fundamental safeguards of system operations and information. WebFOCUS supports a variety of methods that implement the use of secure user account and password policies.

At a minimum, user accounts require password protection, and the passwords assigned to them must be adequately secured and frequently updated. Passwords should be complex enough to make them difficult for unauthorized individuals or entities to guess or generate independently.

User accounts should be suspended after multiple sign-in attempts based on outdated or invalid credentials occur, or whenever passwords expire. Users should be prompted to update expired passwords upon their first attempt to sign in, or to contact an administrator to reset their expired password before attempting to sign in and update it.

WebFOCUS includes the tools and capabilities to incorporate these user account protections on the Internal page of the Administration Console Security tab. However, many organizations prefer to maintain user accounts in an external centralized repository that is accessible to applications across the enterprise. For organizations that manage their users in this way, WebFOCUS can connect to this centralized repository and rely on it for authentication, authorization, and user account management. WebFOCUS provides the configuration for these external connections on the External page of the Administration Console Security tab.

In external authentication, WebFOCUS authenticates users in an external source, such as a Microsoft Active Directory (AD), a Lightweight Directory Access Protocol (LDAP) directory, or a relational database management system (RDBMS) table.

In pre-authentication, WebFOCUS trusts the authentication performed by another system, such as a web server, an Internet identity provider, a Web Access Management system, or another application.

External authentication and pre-authentication providers offer the advantage of centralized user account maintenance and conform to the user account maintenance standards of the enterprise. They enable administrators to focus on the core tasks of administering the WebFOCUS application, and conform to the user account maintenance standards of their organization without any further effort.

If your installation of WebFOCUS relies upon internal authentication, you must establish sign-in and password policies on the Internal Security page of the Administration Console. These policies protect WebFOCUS by limiting the number of sign-in attempts and by limiting the period of time in which a password is valid. They also enable you to establish an appropriate level of complexity and length for user passwords and limit the frequency with which a password can be reused. WebFOCUS stores internal passwords securely using the SHA-2 Secure Hash Algorithm.

Best Practice

To establish secure authentication and user account policies, we recommend the following:

1. If your organization uses a centralized repository, such as Microsoft Active Directory or an LDAP repository, we recommend that you use that external repository for pre-authentication or external authentication, based on the requirements of your application.
2. However, if your organization does not use a centralized authentication provider, use the internal authentication features provided to you within WebFOCUS and ensure that they:
 - ☐ Conform to user account policies established in your organization.
 - ☐ Provide the maximum protection against unauthorized user access.
 - ☐ Require the minimum amount of manual maintenance and review.

Additional Resources

For more information about user authentication or internal user account and password protection, see the *TIBCO WebFOCUS® Security and Administration* technical content.

Use TIBCO WebFOCUS Resource Templates and TIBCO WebFOCUS Reporting Server Access Control Templates

This best practice describes how to use WebFOCUS Resource Templates and Reporting Server Access Control Templates to ensure that consistent and predictable levels of security access are maintained in all workspaces and associated resources. It can be applied by a WebFOCUS administrator.

In this chapter:

- ☐ [Overview](#)
- ☐ [Best Practice](#)
- ☐ [Additional Resources](#)

Overview

The WebFOCUS security model offers administrators the flexibility to establish complex security policies. However, many organizations find that their security needs can be met by a small number of standard user roles and a straightforward pattern of access rights. WebFOCUS provides resource templates for enterprises with departments or divisions, and SaaS providers who require both common and tenant-specific reporting resources when creating workspaces.

Several resource templates are provided that will create folders, groups, roles, and rules for typical enterprise or SaaS deployments. You can implement these templates as is, or adapt them to your requirements. You can also develop custom resource templates for your organization.

Custom resource templates are adaptations of standard resource templates designed to support a unique set of business or operating requirements. Workspaces created from custom resource templates conform to a specialized combination of rules and resources. Custom Resource Templates help ensure that the privileges and resources required by a particular group or activity are automatically built into all of the workspaces created for their support.

Reporting Server Access Control Templates, in conjunction with WebFOCUS Resource Templates, provide a comprehensive access control solution for users in an Enterprise or SaaS Tenant deployment of WebFOCUS.

Reporting Server Access Control Templates are configurations of groups, roles, and privileges. When defined on a Reporting Server, they automatically grant users access to the application directories and capabilities of that server at a level of access that is appropriate to the group and role to which they are assigned.

For example, a user assigned only to the WebFOCUS Marketing/AdvancedUsers group can create reports using metadata residing in the marketing application directory, but not metadata residing in the finance application directory. Another user, assigned to the WebFOCUS Marketing/Developers group, can access Reporting Server Console tools to monitor their connections and agents, while other marketing users who are not in that group cannot.

If you only need to support a small number of groups, you can use the Reporting Server Console to create individual application directories manually, and then configure access privileges for each one. However, when there is a pattern of access between group names and application directories, the implementation of server access control templates is a best practice that saves time, imposes consistency on the results, and is easy to use. Server access control templates allow for the best integration of WebFOCUS resource templates with application directory access privileges and the assignment of users to their proper server role.

Best Practice

We recommend that you use WebFOCUS Resource Templates in combination with Reporting Server Access Control Templates to manage workspace development.

Additional Resources

For more information about access control templates or custom resource templates, see the *TIBCO WebFOCUS® Security and Administration* technical content.



Chapter 8

Build Secure Applications

This best practice addresses how to use application development tools to build applications that protect against web vulnerabilities. It can be applied by a WebFOCUS administrator or an Application Developer.

In this chapter:

- ☐ [Overview](#)
 - ☐ [Best Practice](#)
 - ☐ [Additional Resources](#)
-

Overview

WebFOCUS contains many built-in protections to defend against Web vulnerabilities. However, the development of applications within WebFOCUS can introduce additional vulnerabilities, and developers need to secure their application code by including such protections as are listed in this document.

1. **Protect against injection flaws.**

Injection flaws allow attackers to execute unintended commands or access unauthorized data. Injection attacks, such as SQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The hostile data from the attacker can trick the interpreter into executing unintended commands or accessing unauthorized data.

WebFOCUS contains a number of functions and filters that protect against the injection of untrusted data. Using these functions and filters, application developers can create a whitelist of valid input variables. Existing WebFOCUS language functions that reject data that does not conform to pre-established formats in all database input operations and subroutines that validate all data must be included in all applications that input data. Administrators and developers must write and deploy these features in WebFOCUS applications.

a. **Add the CHKFMT function to all applications that include character string database input operations.**

The CHKFMT function checks a character string for incorrect characters or character types. It compares each character string to a second string, called a mask, by comparing each character in the first string to the corresponding character in the mask.

- b. **Add the CHKNUM function to all applications that include numeric string database input operations.**

The CHKNUM function checks a character string for numeric format. If the string contains a valid numeric format, CHKNUM returns the value 1. If the string contains characters that are not valid in a number, CHKNUM returns zero (0).

- c. **Add the GETTOK function to all applications that include database input operations that extract character or numeric strings from larger data for validation.**

The GETTOK function divides a character string into substrings, called tokens. The data must have a specific character, called a delimiter, that occurs in the string and separates the string into tokens. GETTOK returns the token specified by the token_number argument. GETTOK ignores leading and trailing blanks in the source character string.

- d. **Add regular expression validation to all database input operations.**

You can validate a parameter value without accessing the data by using the REGEX mask. The REGEX mask specifies a regular expression to be used as the validation string. A regular expression is a sequence of special characters and literal characters that you can combine to form a search pattern.

A sample -SET command with a REGEX mask is shown in the following example:

```
-SET &VAR=&VAR. ( | VALIDATE=REGEX,REGEX='^[a-zA-Z0-9, @]+$' ) . ;
```

This command allows the variable &VAR to accept values that include alphanumeric characters, commas (,) and the at sign (@). (The +\$ argument is a modifier.)

Examples of values that will pass this validation include:

```
Smith,Robert
RobertSmith123
RobertSmith@345
```

Examples of values that will not pass this validation include:

- ☐ RobertSmith@ibi.com, which contains a period (.).
- ☐ RobertO'Reily, which contains an apostrophe (').
- ☐ Robert_Smith, which contains an underscore (_).

2. **Protect against Cross-Site Scripting attacks.**

Cross-Site Scripting (XSS) flaws occur whenever an application takes un-trusted data and sends it to a web browser without proper validation and escaping. Cross-Site Scripting flaws allow attackers to execute scripts in the browser of the victim that can hijack user sessions, deface web sites, or redirect users to malicious sites.

WebFOCUS includes the following Reporting Server configuration settings to protect against Cross-Site Scripting (XSS) attacks, Enable HTMLENCODE, and DEFECHEO. We recommend that you activate both settings on the Reporting Server to activate their protection.

The WebFOCUS language contains the following subroutines that protect against Cross-Site Scripting attacks. Administrators and developers must write and deploy these features in WebFOCUS applications.

- a. **Add the CHKFMT function to all applications that include character string database input operations.**

The CHKFMT function checks a character string for incorrect characters or character types. It compares each character string to a second string, called a mask, by comparing each character in the first string to the corresponding character in the mask.

- b. **Add the CHKNUM function to all applications that include numeric string database input operations.**

The CHKNUM function checks a character string for numeric format. If the string contains a valid numeric format, CHKNUM returns the value 1. If the string contains characters that are not valid in a number, CHKNUM returns zero (0).

- c. **Add the GETTOK function to all applications that include database input operations that extract character or numeric strings from larger data for validation.**

The GETTOK function divides a character string into substrings, called tokens. The data must have a specific character, called a delimiter, that occurs in the string and separates the string into tokens. GETTOK returns the token specified by the token_number argument. GETTOK ignores leading and trailing blanks in the source character string.

- d. **Add the XMLENCOD function to all numeric and character string database input operations.**

This function has a specific relevance for cross-site scripting protections. It neutralizes data returned to the user.

The XMLENCOD function encodes the following five standard characters when they are encountered in a string:

| Character Name | Character Encoded Representation |
|-----------------------|----------------------------------|
| ampersand & | & |
| greater than symbol > | > |

| Character Name | Character Encoded Representation |
|-------------------------|----------------------------------|
| less than symbol < | < |
| double quotation mark " | " |
| single quotation mark ' | ' |

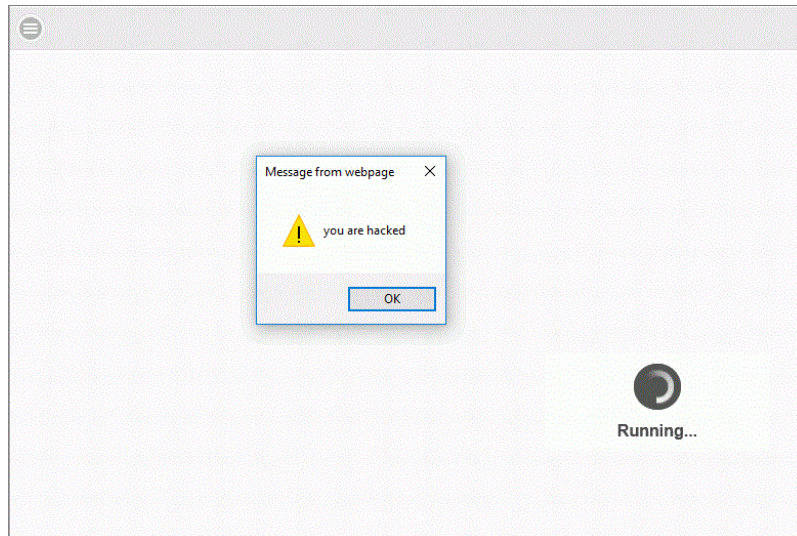
For example, the following procedure does not contain a command that encodes special characters, and therefore, it does not convert any special characters included in a search query into literal characters:

```
-HTMLFORM BEGIN
<HTML>
Hello &VAR.Enter your Name:.
</HTML>
-HTMLFORM END
```

When you enter the search term, `<script>alert('you are hacked')</script>`, as shown in the following image:

The screenshot shows a web application interface. At the top, there are four buttons: a back arrow, a refresh/circular arrow, a magnifying glass, and a circular arrow with a plus sign. Below these buttons is a section titled "Filter Values". Inside this section, there is a label "Enter your Name:" followed by a text input field containing the text `<script>alert('you are hacked')</script>`. To the right of the input field, there is a blue circular icon with three horizontal lines. Further to the right, there are two numbered instructions: "1. Specify values for all parameters." and "2. Select the run button to submit the request."

There is nothing in the procedure to prevent it from executing the script command and displaying the alert, as shown in the following image.



To address this issue, you could add an XMLENCOD function that encodes standard characters within a specified string as literals.

For example, an updated version of the previous procedure is shown in the following example:

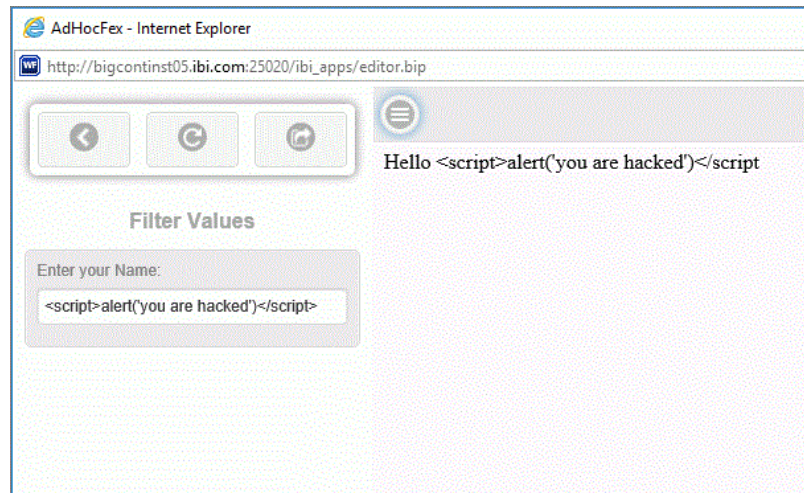
```
-SET &VAR=XMLENCOD(&VAR.LENGTH,&VAR,0,60,'A60');
-HTMLFORM BEGIN
<HTML>
Hello &VAR.Enter your Name:.
</HTML>
-HTMLFORM END
```

This procedure includes the -SET command with an XMLENCOD function.

```
-SET &VAR=XMLENCOD(&VAR.LENGTH,&VAR,0,60,'A60');
```

This function converts the greater than sign (>) and the less than sign (<), which define the script tags within the search term text string and identify it as an executable command, into the literals > and < respectively.

This protected version of the procedure does not execute the script. Instead, it converts it into a plain text response, as shown in the following image.



```
<HTML>
Hello &lt;script&gt;alert(&apos;you are hacked&apos;)&lt;/script
</HTML>
<!--
```

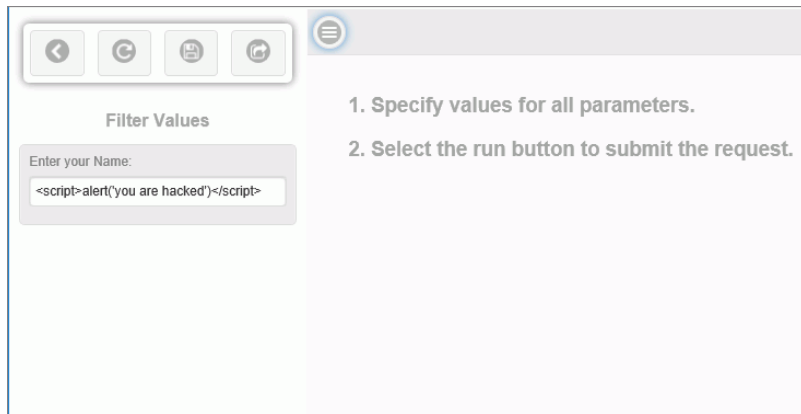
e. **Add Regular Expression validation to all database input operations.**

You can validate a parameter value without accessing the data by using the REGEX mask. The REGEX mask specifies a regular expression to be used as the validation string. A regular expression is a sequence of special characters and literal characters that you can combine to form a search pattern.

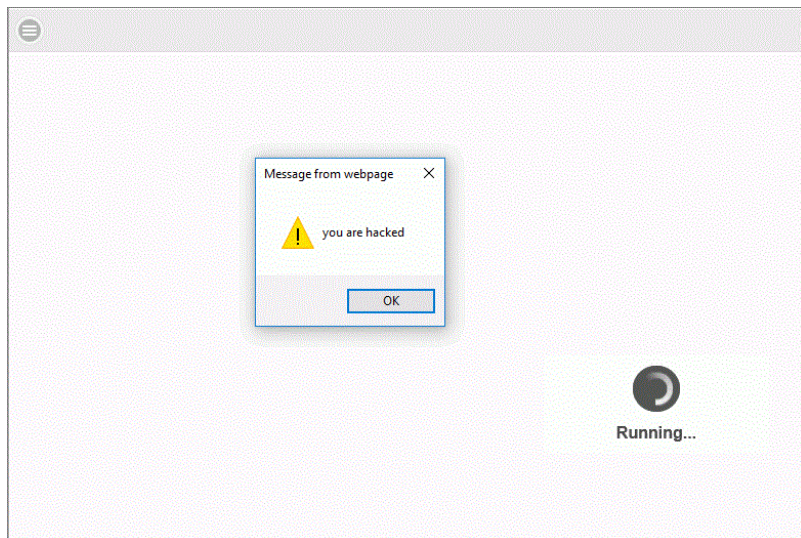
For example, the following procedure does not contain a command that validates regular expressions, and therefore, it does not reject any search query that contains characters that could cause a text string to be interpreted as an executable code statement:

```
-HTMLFORM BEGIN
<HTML>
Hello &VAR.Enter your Name:.
</HTML>
-HTMLFORM END
```

When you enter the search term, `<script>alert('you are hacked')</script>`, as shown in the following image:



There is nothing in the procedure to prevent it from executing the script command and displaying the alert, as shown in the following image.



To address this vulnerability, you could add a -SET command that validates input against a REGEX mask. For example, an updated version of the previous procedure is shown in the following example:

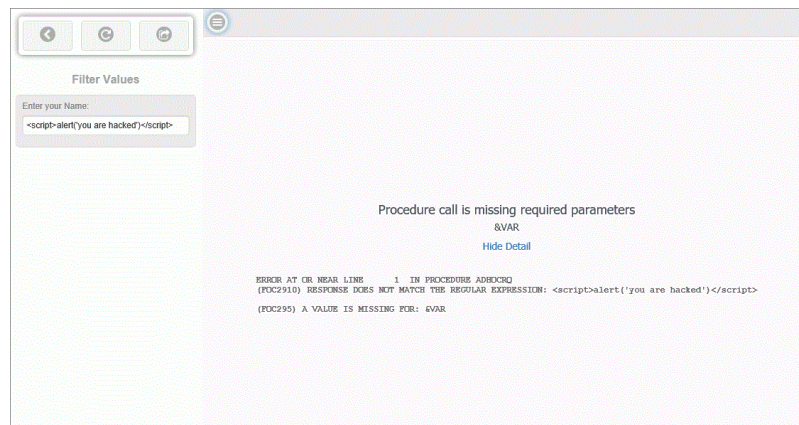
```
-SET &VAR=&VAR.( | VALIDATE=REGEX,REGEX='^[a-zA-Z0-9, @]+$' ). ;
-HTMLFORM BEGIN
<HTML>
Hello &VAR.Enter your Name:.
</HTML>
-HTMLFORM END
```

This procedure includes the -SET command with a REGEX mask.

```
-SET &VAR=&VAR.( | VALIDATE=REGEX,REGEX='^[a-zA-Z0-9, @]+$' ). ;
```

This command allows the variable &VAR to accept search terms containing values that include alphanumeric characters, commas (,) and the at sign (@), but reject any search term that contains values outside of the range defined by the REGEX mask.

This protected version of the procedure does not execute the script. Instead, it produces the Response does not match the Regular Expression error message when it is run with a data string that contains invalid characters, as shown in the following image.



We recommend that application developers include these subroutines in every operation that introduces data into a database.

3. Configure the application to neutralize HTML tags within untrusted data in the database.

This best practice protects the database and the WebFOCUS application against data that could contain a stored cross-site scripting attack.

WebFOCUS contains the following feature that prevents the use of untrusted data:

Enable `HTMLENCODE=ON`

When this setting is on, WebFOCUS encodes HTML tags within data as plain text so that the browser does not interpret them as executable code. It disables the rendering of HTML tags within a browser when these tags are stored within the actual data, or created using a `DEFINE` or `COMPUTE` command.

This practice prevents an attack on the server resulting from the insertion of executable code into data.

4. **Protect against Cross-Site Request Forgery**

Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which he or she is currently authenticated. To address this vulnerability, include a unique Cross-Site Request Forgery token in a hidden field of sensitive HTTP POST Requests. This token causes the value to be sent in the body of the HTTP POST request.

WebFOCUS includes a Cross-Site Request Forgery token in its configuration, and the setting that activates it, Cross Site Request Forgery Protection (`IBI_CSRF_ENFORCE`), is activated by default. However, Cross-Site Request Forgery tokens are not included in applications you develop from WebFOCUS. You must include them in the application code.

For example, to include references to CSRF token variables in -HTMLFORM Dialogue Manager Procedures, there are two requirements. First, the CSRF Token Name and CSRF Token Value variables must be added to the client side logic by assigning them to the `site.wfs` file. To do so, type the following values in the Custom Settings page of the Administration Console, and save the updated page.

```
<SET>IBI_CSRF_Token_Name(PASS)
<SET>IBI_CSRF_Token_Value(PASS)
```


Second, references to these variables must be added to the -HTMLFORM BEGIN/END section of each -HTMLFORM Dialogue Manager procedure that will use them, as shown in the following example.

```
-HTMLFORM BEGIN
<body onload="document.form.submit()">
<form name=form id=form action="/ibi_apps/run/ibfs" method="post">
<input type="hidden" name="IBFS_path" value="/WFC/Repository/sales/
salesbyregion.fex" />
<input type="hidden" name="IBFS_action" value="run" />
<input type="text" name="COUNTRY" value="ITALY" />
<input type="hidden" name="|IBI.AMP.IBI_CSRF-Token_Name;" value="|
IBI.AMP.IBI_CSRF-Token_Value;" />
</form>
-HTMLFORM END
```

Note: This sequence is taken from a Release 8.2.01M focexec procedure. The URL in the action attribute of the form tag is not supported in WebFOCUS Release 8.2.01 and prior releases.

5. Protect against Click-jacking attacks.

Click-jacking occurs when an attacker uses multiple transparent or opaque layers over a webpage to trick a user into clicking on a button or link that affects a different page than the page they intended to affect. Keystrokes typed on such a webpage can be also hijacked to a different page. As a result, users who believe they are working with a trusted and legitimate web page are actually working with a malicious web page.

By causing users to activate malicious code or script by clicking on what appears to be a legitimate button or other webpage component, click-jacking can cause users to expose confidential information or transfer control of their computer to a malicious entity. To ensure that all WebFOCUS interfaces windows are top level windows, use the Cross-Origins Settings dialog box and employ code to configure the use of framing in other workspaces.

6. Protect against sensitive information exposure.

By limiting the level of detailed error information, you prevent the potential disclosure of sensitive information to end users.

By removing detailed system information from WebFOCUS Client error messages, you prevent the potential disclosure of specific error message details, error descriptions, and other information that a malicious party could extract and use to launch additional attacks. Administrators can replace these detailed messages with error messages that provide users with a simplified text message that conveys relevant information.

To do so:

- a. Limit the level of information included in error messages by assigning the value of None to the Message Detail (IBI_MESSAGE_DETAIL) setting.

- b. Select the Reporting Server UINFO privilege to minimize the level of detail displayed in error messages the Reporting Server produces.

7. Incorporate security protections into the application development process.

Malicious code or malware inadvertently included in new applications can cause them to expose confidential information, execute unwanted or unauthorized commands, or introduce worms or other malware on the computers with which they interact.

To protect against these intrusions, you must implement your own Security Software Development Lifecycle (SDLC) that introduces code review, Dynamic Application Security Testing (DAST), and ethical hacking techniques into your application development process. You should also know and understand the OWASP Top 10 security issues and the Payment Card Industry Data Security Standards, because they represent generally accepted standards that are useful guidelines for your own software development process.

Be sure to have the Security team inspect all new and updated application code for vulnerabilities from the outset of the WebFOCUS application development process. Code review must not be delayed until development is complete.

The SDLC is incorporated into the WebFOCUS development process, including the following activities:

- ☐ SAST - Static Application Security Testing
- ☐ DAST - Dynamic Application Security Testing
- ☐ Ethical Hacking
- ☐ Independent Internal Security Audits from the Security Engineering team, an independent Internal Testing Team of TIBCO.
- ☐ Prioritization of Customer Security Flaws

When developing WebFOCUS applications, you need to implement your own SDLC, including the following activities, from the outset of your development process:

- ☐ Code Review
- ☐ DAST - Dynamic Application Security Testing
- ☐ Ethical Hacking
- ☐ Security Audits, via an internal Information Security Team or a third party.

Best Practice

We recommend that application developers incorporate these recommendations, based on their application requirements. It is possible that not all of the recommended settings can be followed, but from an application development process, Developers can implement these coding techniques, some settings, and the SDLC process to protect against the attacks described in this section.

Additional Resources

For more information about configuration and implementation topics, see the *TIBCO WebFOCUS® Security and Administration* technical content, and the *TIBCO WebFOCUS® Reporting Server Administration* technical content. For more information about application development topics, see the *TIBCO WebFOCUS® Creating Reports With TIBCO WebFOCUS® Language* technical content, the *TIBCO WebFOCUS® Developing Reporting Applications* technical content, the *TIBCO WebFOCUS® Describing Data With TIBCO WebFOCUS® Language* technical content, and the *TIBCO WebFOCUS® Using Functions* technical content.

Secure Apache Solr Communications

This best practice addresses the need to secure connections between WebFOCUS and Apache Solr to ensure the confidentiality, authenticity, and integrity of all communications. It also addresses the need to ensure that indexing and search operations affect only those content resources and domains to which the user issuing the indexing operation or search query has been granted access. It can be applied by a WebFOCUS administrator and requires the support of an Apache Solr administrator.

In this chapter:

- ☐ [Overview](#)
 - ☐ [Best Practice](#)
 - ☐ [Additional Resources](#)
-

Overview

In the standard configuration of WebFOCUS, search queries and indexing operations are directed to an Apache Solr application that uses a server based on Apache Lucene, a Java-based open-source information retrieval product that supports search operations.

Apache Solr is independent of the applications it supports. It runs in an external environment and can support search queries from multiple users, whose access privileges can vary from unrestricted access to tightly restricted access within a single application. The same Apache Solr server can also support search and index operations from multiple applications that are not at all associated with WebFOCUS. In this multi-user, multi-tenant environment, administrators have an obligation to ensure that the results of search and index operations are available only to the application and user from which they originated.

Communications move between WebFOCUS and Apache Solr in the form of URLs that include query request information and results returned in the form of JSON files or delimited files using a different format.

An Apache Solr server is included in the on-premises product installation, by default. An Apache Solr Server is available to cloud installations, but it is not hosted within the same environment. Therefore, connectivity to a remote server and all of the protections that are required to secure such connections are advised.

Because they must move across the secure barriers of the WebFOCUS environment to those of the Apache Solr application, queries and index operations can be vulnerable to man-in-the-middle attacks that produce compromised results and provide an entry point for unauthorized individuals to manipulate the systems they target.

Even though communications between WebFOCUS and Apache Solr can be unrestricted, given the need to secure these vulnerable communications, WebFOCUS requires the use of Basic Authentication within the Apache Solr configuration for all customers using the cloud installation and recommends it for customers using an on-premise installation.

WebFOCUS supports the use of Solr Basic Authentication by providing settings on the Search Settings page that contain the credentials provided to you by your Apache Solr administrator. It also enables you to specify a target URL using the HTTPS protocol to support TLS (SSL) encryption. Finally, it ensures that your query and index operations remain unavailable to users outside of your organization by enabling you to create a unique name for your index collection and establish it within WebFOCUS and Apache Solr.

Note that, in addition to Basic Authentication, Apache Solr also supports Authorization, Audit Logging, and IP Access control within its Security configuration. The use of these additional features is beyond the scope of WebFOCUS requirements, and other than establishing the use of Basic Authentication and the HTTPS protocol, WebFOCUS does not contain configuration settings that support these additional security features in Apache Solr. If your organization requires you to support these additional features, contact your organization's Apache Solr administrator to learn more.

Best Practice

We recommend that you configure Apache Solr to use Basic Authentication by taking the following steps.

1. Work with your Apache Solr administrator to develop Apache Solr Basic Authentication, or if you will manage both applications, configure it on your own.
2. Add the following Basic Authentication values to the WebFOCUS Configuration:
 - a. Add a User ID to the *User Name for Basic Authentication (IBI_SEARCH_USERNAME)* setting.
 - b. Replace the default Password provided in the *Password for Basic Authentication (IBI_SEARCH_PASSWORD)* setting with a unique password for your organization.

When Solr Basic Authentication is in use, a unique search collection is also required. Therefore, we recommend that you:

1. Replace *ibi-protected*, the default value that appears in the *Collection Name (IBI_SEARCH_COLLECTION)* setting, with a collection name that is unique to your organization.

2. Assign the same unique name to your collection in the Solr Collections API.

For more information, see the *Collections/Core Admin* topic and the *Collections API* topic in the *Apache Solr Reference Guide*.

We also recommend that you extend the use of an HTTP Strict Transport Security (HSTS) policy to all communications between the WebFOCUS Client and the Solr Server.

1. Ensure that the Apache Solr configuration also requires the use of SSL/TLS communications.
2. Create a Self-Signed Certificate and assign it to your configuration of TIBCO WebFOCUS.
 - a. If you are using Apache Tomcat as your web server, follow the procedures in the *Configuring TIBCO WebFOCUS for SSL* topics, which are located in the *TIBCO WebFOCUS® Security and Administration* technical content.
 - b. If you are using a web server from a different vendor, follow the instructions describing how to configure SSL/TLS for that server provided by your vendor.
3. Assign a URL that includes the HTTPS protocol to the *Solr URL* (*IBI_INFOSEARCH_SOLR_URL*) setting.

Additional Resources

For more information, see *Configuring Solr Basic Authentication* in the *TIBCO WebFOCUS® Security and Administration* technical content.

For more information about Apache Solr Security, see the *Securing Solr* section of the *Apache Solr Reference Guide* for the version of Apache Solr installed in your organization.

Glossary

Administration Console

The interface that administrators use to manage the WebFOCUS environment and configuration settings.

Administration privileges

System administrator privileges that are generally only assigned to WebFOCUS administrators.

Advanced Reporting privileges

Privileges that can be assigned to users who need to create and share their own reports, generally granted as a supplement to the Basic Reporting privileges.

alternate zone

The security zones that defines secondary authentication methods to be used based on the user network location.

anonymous access

Unauthenticated access to resources.

Application Development privileges

Privileges that can be assigned to developers so they can create complete WebFOCUS applications using only web-based tools.

auditing

The process of tracking user access to tools and resources and logging important administrative actions.

authentication

The process of confirming the identity of a user.

authorization

The process of enforcing user privileges to control the access to resources and tools within an application.

AUTOADD

The process of automatically adding pre-authenticated and externally authenticated users to WebFOCUS, if the user accounts exist in the external source, but do not already exist in WebFOCUS.

CHKFMT

The CHKFMT function checks a character string for incorrect characters or character types. It compares each character string to a second string, called a mask, by comparing each character in the first string to the corresponding character in the mask.

CHKNUM

The CHKNUM function checks a character string for numeric format. If the string contains a valid numeric format, CHKNUM returns the value 1. If the string contains characters that are not valid in a number, CHKNUM returns zero (0).

clickjacking

A web attack that nests content inside frames to trick users into clicking on a button or link and unknowingly enabling a malicious action. Clickjacking is also known as a UI redress attack.

Cross-Site Request Forgery (CSRF)

A web attack that injects malicious scripts into trusted web sites.

Cross-Site Scripting

Cross-Site Scripting (XSS) flaws occur whenever an application takes un-trusted data and sends it to a web browser without proper validation and escaping. Cross-Site Scripting flaws allow attackers to execute scripts in the browser of the victim that can hijack user sessions, deface web sites, or redirect users to malicious sites.

DEFECHO

When activated, the DEFECHO setting causes the Reporting Server to display procedure code while an application is running.

Dynamic DBA Security

Protects the repository database from incursion by unauthorized users. For example, you can limit access to a given data source by user name, and within that privilege establish limits on the ability of a specified user to read, write, or update data. You can also restrict a user to certain fields or segments of data and ensure that only records that pass a validation test are retrieved.

| | |
|--------------------------------|---|
| external authentication | The process of confirming the identity of a user through an application other than WebFOCUS. |
| folder | A container for repository content. |
| GETOK | The GETTOK function divides a character string into substrings, called tokens. The data must have a specific character, called a delimiter, that occurs in the string and separates the string into tokens. GETTOK returns the token specified by the token_number argument. GETTOK ignores leading and trailing blanks in the source character string. |
| group | A collection of users or subgroups which require similar capabilities or access to the same resources. |
| HTMLENCOD | When this setting is on, WebFOCUS encodes HTML tags within data as plain text so that the browser does not interpret them as executable code. This practice prevents an attack on the server resulting from the insertion of executable code into data. |
| https | Hypertext Transfer Protocol over Secure Socket Layer (https) technology, which includes the Transport Layer Security (TLS) protocol previously known as Secure Socket Layer (SSL), establishes a secure connection between users and the WebFOCUS client. |
| IBFS | A logical addressing system used by WebFOCUS to store and retrieve objects. Every object has a unique IBFS path. |

injection flaws

Injection flaws allow attackers to execute unintended commands or access unauthorized data. Injection attacks, such as SQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The hostile data from the attacker can trick the interpreter into executing unintended commands or accessing unauthorized data.

Lightweight Directory Access Protocol (LDAP)

An application protocol for accessing and maintaining distributed directory information services over an Internet Protocol (IP) network. WebFOCUS can be configured to use LDAP for external authentication or authorization.

OWASP

Open Web Application Security Project. An online community that publishes security articles, methodologies, and establishes the OWASP Top Ten Most Critical Web Application Security Risks.

pre-authentication

WebFOCUS authentication relying on authentication already performed by a third-party security provider.

PTH

(Process Table Handler) The default security provider used when the WebFOCUS Reporting Server is first installed. Users are authenticated against a list maintained in the admin.cfg file.

public access

Supports anonymous or unauthenticated access to resources in the WFC/Repository/Public folder, as well as to procedures on the WebFOCUS Reporting Server.

RDBMS

A relational database management system (RDBMS) table can be used to support external authentication.

| | |
|--|--|
| REGEX | The REGEX mask specifies a regular expression to be used as the validation string. A regular expression is a sequence of special characters and literal characters that you can combine to form a search pattern. |
| resource | Any folder, item, library content, portal, privilege, report procedure, role, user, or group to which access can be controlled or to whom abilities can be granted. |
| resource template | A template that creates folders, rules, roles, and portals based on an existing model. For example, an enterprise workspace template or a SaaS tenant workspace template. |
| SAST | Static Application Security Testing |
| SDLC | Software Development Lifecycle. A sequence of distinct stages for software design and development. |
| Security Center | The interface used to manage users, groups, roles, and rules. This management ability may be delegated to users or groups without full administrative control. |
| security provider | Authenticates and authorizes users. |
| Server Access Control Templates | Configurations of groups, roles, and privileges that, when defined on a Reporting Server, automatically grant users in those groups an appropriate level of access to the application directories and capabilities available on that server. |

service account

An account used to execute services or perform system tasks that should not be performed by an individual user account.

service provider

In a SaaS deployment, the organization which implements WebFOCUS and grants controlled access to its clients, tenant users.

session privilege

A privilege that is identified by WebFOCUS during sign-in and then cached for the duration of the session.

shared resource

A private resource that an individual user has shared with other users or groups. Shared resources are made available to users through the Shared Content folder.

single sign on (SSO)

Access to multiple related, but independent, software systems.

Software as a Service (SaaS)

A business model in which an organization provides controlled access to their services to multiple clients.

SQL Passthru

A capability that allows WebFOCUS users to execute SQL Passthru commands that would affect the database on the Reporting Server.

SSL

Secure Socket Layer. A component of https technology.

subsystem

A component in the hierarchy that organizes objects in the WebFOCUS Repository.

| | |
|------------------|---|
| superuser | An account with the special privileges needed to administer and maintain the system. Superuser access overrides all other security rules. |
| tenant | In a SaaS deployment, a client organization which is granted controlled access by the client service organization that implements WebFOCUS. In a multi-tenancy deployment, although tenant users belong to the EVERYONE group, along with the service provider users, the tenant users are only aware of other users within their own organization. |
| title | The display name of an object. |
| TLS | Transport Layer Security. An updated version of SSL. A component of https technology. |
| UINFO | A Reporting Server setting that disables the display of My Console, Error Message, Logon Info, Server Version, Console Log and Help. |
| XMLENCOD | A function encodes the following five standard characters when they are encountered in a string, ampersand (&), greater than symbol (>), less than symbol (<), double quotation mark ("), and single quotation mark ('). |

Legal and Third-Party Notices

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document contains confidential information that is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, FOCUS, iWay, Omni-Gen, Omni-HealthData, and WebFOCUS are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle Corporation and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2022. TIBCO Software Inc. All Rights Reserved. TIBCO Confidential Information.

Index

A

Agile best practices [7](#)
Automatic Sign-out [23](#)

C

CHKFMT [31](#)
CHKNUM [32](#)
CI/CD [7](#)
Click-Jacking [10](#), [40](#)
Code Development [7](#)
Code Review [5](#)
Continuous Integration/Continuous Development
[7](#)
Cross-Site Request Forgery [10](#), [39](#)
Cross-Site Scripting [32](#)

D

DAST [6](#)
Data at Rest [18](#)
default credentials [21](#)
DEFECHO [17](#)
Dependency-Check [6](#)
Dynamic Application Security Testing [6](#)
Dynamic DBA Security [18](#)

E

error message detail [23](#), [40](#)
ethical hacking [6](#)

F

Formal QA [7](#)
Functional Tests [7](#)

G

GETTOK [32](#)

H

HTMLENCOD [39](#)
HTMLENCODE [17](#)
https [25](#)

I

IAST [6](#)
Information Assurance [5](#)
injection flaws [31](#)
Interactive Application Security Testing [6](#)
Inverse Reinforcement Learning [6](#)
IP Restriction filtering [18](#)
IRL [6](#)

J

JSESSIONID [21](#)

L

LDAP [28](#)

M

Microsoft Active Directory [28](#)

N

National Vulnerability Database [6](#)

O

OWASP [6](#)

P

Product Build [7](#)

Production Environment Deployment [8](#)

public access [22](#)

R

RDBMS [28](#)

REGEX [36](#)

Resource Templates [10](#), [29](#)

S

SAST [5](#)

SDLC [5](#), [41](#)

Secure Applications [31](#)

Secure Authentication [27](#)

Security Audits [6](#)

Security Bugs

Security Bugs (*continued*)

Prioritization [6](#)

Security Training [6](#)

Server Access Control Templates [10](#), [29](#)

Server Administrator Password [21](#)

session timeout [23](#)

Software Release [8](#)

SQL Passthru [16](#)

SSL [25](#)

Staging Environment Deployment [7](#)

Static Application Security Testing [5](#)

T

TLS [25](#)

U

Unit Tests [7](#)

useHttpOnly [21](#)

User Account Policies [27](#)

W

WebFOCUS Client Settings [9](#), [21](#)

WebFOCUS infrastructure [8](#), [11](#)

WebFOCUS Server Settings [9](#), [15](#)

X

XMLENCOD [33](#)