NetScaler Advanced Security Administration

CNS-318-1I

Lab Guide

Credits Page

Title	Name
Architects	Jesse Wilson Howard Weise
Product Managers	Lissette Jimenez Matthew Brooks
Technical Solutions Developers	Anton Mayers Aman Sharma Rhonda Rowland
Instructional Designer	Elizabeth Diaz
Graphics Designers	Ryan Flowers Joe Baum
Publication Services	Akhilesh Karanth Rahul Mohandas Zahid Baig
Special Thanks	Layna Hurst Todd Hurst Layer 8 Training

Contents

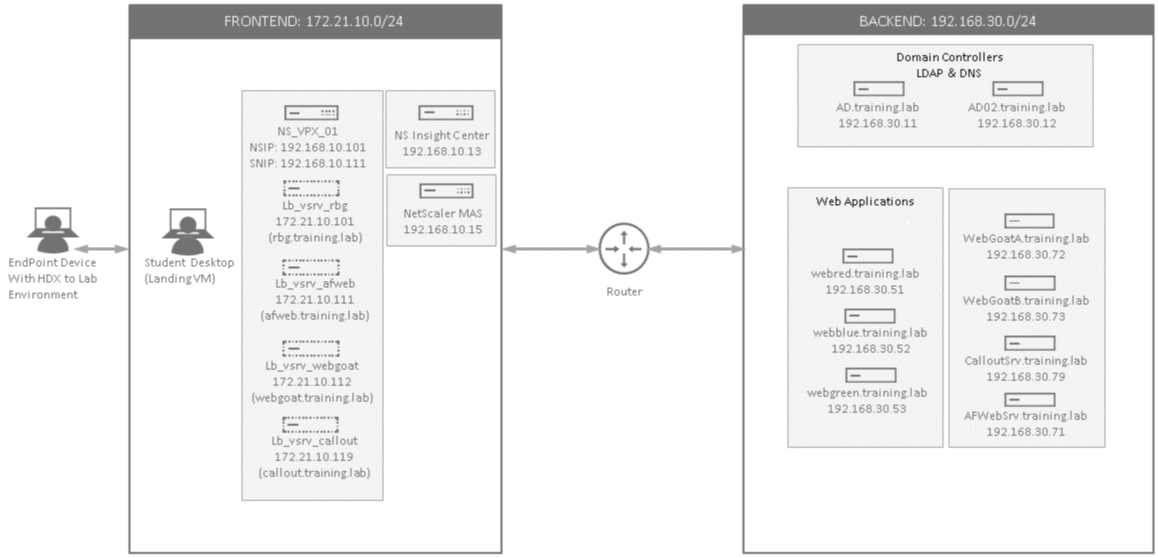
Lab Guide Overview

In this lab guide, you will get valuable hands-on experience with NetScaler and its security features including Application Firewall. This lab guide will enable you to work with product components and perform required steps for configuration of the NetScaler for web application security.

Lab Environment Overview

Lab Diagram

LAB DIAGRAM



SERVER LIST

Virtual Machine Name	Domain FQDN	IP Address	Description
AD.training.lab	ad.training.lab	192.168.30.11	Domain Controller (training.lab)
AD02.training.lab	ad02.training.lab	192.168.30.12	Domain Controller 2 (training.lab)
WebRed	webred.training.lab	192.168.30.51	Web Server
WebBlue	webblue.training.lab	192.168.30.52	Web Server
WebGreen	webgreen.training.lab	192.168.30.53	Web Server
AFWebSrv	afwebsrv.training.lab	192.168.30.71	Web Server - Application Firewall Test App
WebGoatA	webgoatA.training.lab	192.168.30.72	Web Server - Application Firewall Test App
WebGoatB	webgoatB.training.lab	192.168.30.73	Web Server - Application Firewall Test App
CalloutSrv	calloutsrv.training.lab	192.168.30.79	Web Server - Blacklist Server / HTTP Callout agent
Student Desktop	--	192.168.10.10	Student lab workstation; landing workstation. All labs performed from this system.

NetScaler List

Virtual Machine Name	NSIP Address	Subnet IP (SNIP) Address	Description
NS_VPX_01	192.168.10.101	SNIP1: 192.168.10.111 (traffic)	NS_VPX_01 is the only NetScaler in this environment. It is already configured with NSIP, SNIP, and initial load balancing virtual servers.
NS_InsightCenter	192.168.10.13		
NetScaler MAS	192.168.10.15		mas.training.lab

CREDENTIALS LIST (1): Training Domain Users and Groups for NetScaler Administration

User Name	Groups	Password	Description
administrator	Domain Admins	Password1	Domain administrator account which can be used to access domain controllers via console or RDP. Otherwise, not needed in class.

Virtual Servers, FQDNs, and VIPs - Days 1-3

Virtual Server Names	FQDN	VIPs	Course
lb_vsrv_rbg	rbg.training.lab	172.21.10.101	CNS318/CNS319
lb_vsrv_afweb	afweb.training.lab	172.21.10.111	CNS318/CNS319
lb_vsrv_webgoat	webgoat.training.lab	172.21.10.112	CNS318/CNS319
lb_vsrv_callout	callout.training.lab	172.21.10.119	CNS318/CNS319

Virtual Servers, FQDNs, and VIPs - Days 4-5

Virtual Server Names	FQDN	VIPs	Course
lb_vsrv_afweb_ssl	afweb.training.lab	172.21.10.111	CNS319
lb_vsrv_webgoat_ssl	webgoat.training.lab	172.21.10.112	CNS319

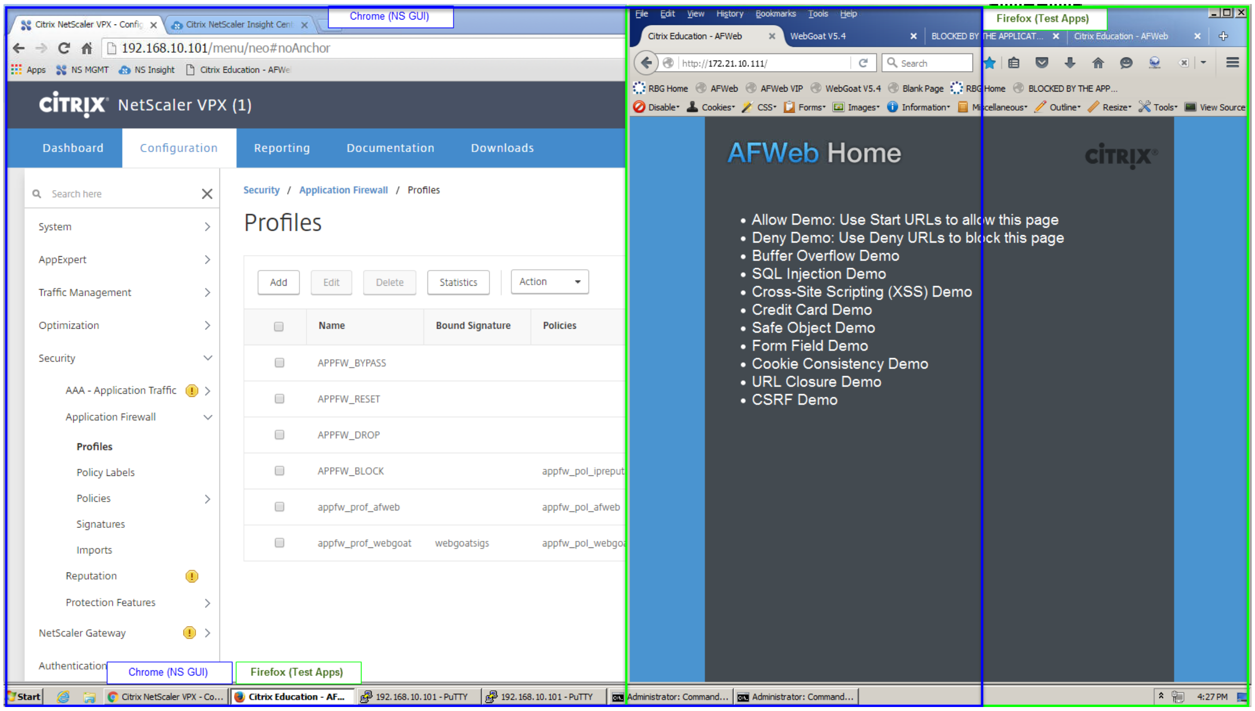
Working with the Labs

NetScaler Configuration and Application Testing

It is strong recommended, when running the exercises in this class, that you perform NetScaler configurations using Chrome web browser to access the NetScaler Configuration Management utility and test application attacks and protections in Firefox.

- This will allow you to switch back-and-forth from the configuration utility to the test application multiple times during each exercise.
- When certain labs require you to reset cookies or the browser's session state it will only affect Firefox and the test applications and not your connection to the management console in Chrome.
- Many of the troubleshooting and test utilities that will be required for the Application Firewall and other exercises are only installed for Firefox.

A suggested windows arrangement is pictured below:



Profiles

Name	Bound Signature	Policies
APPFW_BYPASS		
APPFW_RESET		
APPFW_DROP		
APPFW_BLOCK		appfw_pol_ipreput
appfw_prof_afweb		appfw_pol_afweb
appfw_prof_webgoat	webgoatsigs	appfw_pol_webgoa

AFWeb Home

- Allow Demo: Use Start URLs to allow this page
- Deny Demo: Use Deny URLs to block this page
- Buffer Overflow Demo
- SQL Injection Demo
- Cross-Site Scripting (XSS) Demo
- Credit Card Demo
- Safe Object Demo
- Form Field Demo
- Cookie Consistency Demo
- URL Closure Demo
- CSRF Demo

During the Application Firewall exercises, the NetScaler Configuration Utility (GUI) will be run in the web browser to perform most of the configuration. You will also be asked to open two separate PuTTY sessions to make SSH connections to the NetScaler CLI.

- **Putty (1)** will be used to view the Syslog output as it is generated, using the following commands:

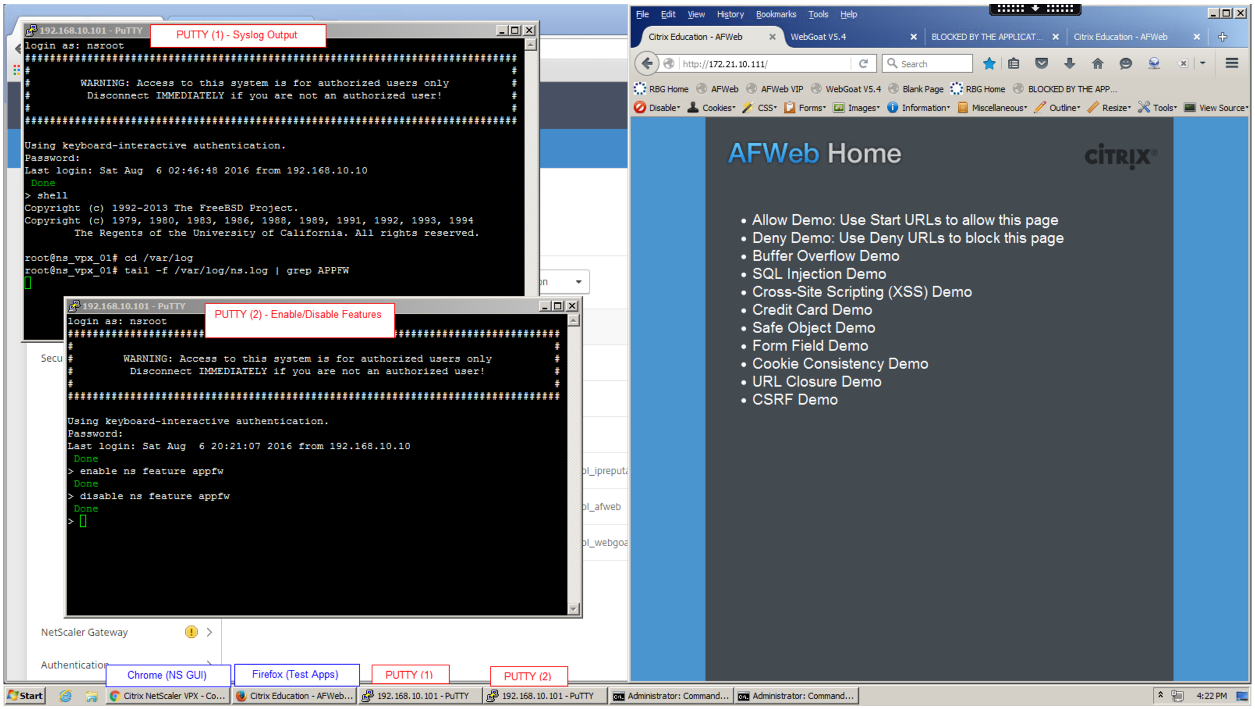
```
shell
cd /var/log/
tail -f /var/log/ns.log | grep APPFW
```

- **Putty (2)** will be used to toggle the Application Firewall feature on or off as required:

```
enable ns feature appfw
disable ns feature appfw
```

These SSH sessions will be used to make it easy to view Application Firewall violations as they occur or to switch the feature on and off frequently during exercises. The lab will instruct you when to create the sessions and when to use them.

It is recommended that students keep the two session running during the Application Firewall labs and switch between the Putty sessions as needed. A suggested arrangement for the windows is displayed below.



```
login as: nsroot
##############################################################################
#                                                                            #
#        WARNING: Access to this system is for authorized users only         #
#         Disconnect IMMEDIATELY if you are not an authorized user!          #
#                                                                            #
##############################################################################

Using keyboard-interactive authentication.
Password:
Last login: Sat Aug  6 02:46:48 2016 from 192.168.10.10
 Done
> shell
Copyright (c) 1992-2013 The FreeBSD Project.
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994
        The Regents of the University of California. All rights reserved.

root@ns_vpx_01# cd /var/log
root@ns_vpx_01# tail -f /var/log/ns.log | grep APPFW
```

```
login as: nsroot
##############################################################################
#                                                                            #
#        WARNING: Access to this system is for authorized users only         #
#         Disconnect IMMEDIATELY if you are not an authorized user!          #
#                                                                            #
##############################################################################

Using keyboard-interactive authentication.
Password:
Last login: Sat Aug  6 20:21:07 2016 from 192.168.10.10
 Done
> enable ns feature appfw
 Done
> disable ns feature appfw
 Done
>
```

- Allow Demo: Use Start URLs to allow this page
- Deny Demo: Use Deny URLs to block this page
- Buffer Overflow Demo
- SQL Injection Demo
- Cross-Site Scripting (XSS) Demo
- Credit Card Demo
- Safe Object Demo
- Form Field Demo
- Cookie Consistency Demo
- URL Closure Demo
- CSRF Demo

Citrix Hands-on Labs

What are Hands-on Labs?

Hands-on Labs from Citrix Education allows you to revisit, relearn, and master the lab exercises covered during the course. This offer gives you 25 days of unlimited lab access to continue your learning experience outside of the classroom.

Claim introductory pricing of $500 for 25 days of access.

Contact your Citrix Education representative or purchase online here.

Why Hands-on Labs?

Practice outside of the classroom

You'll receive a fresh set of labs, giving you the opportunity to recreate and master each step in the lab exercises.

Test before implementing

Whether you're migrating to a new version of a product or discovered a product feature you previously didn't know about, you can test it out in a safe sandbox environment before putting in live production.

25 days of access

Get unlimited access to the labs for 25 days after you launch, giving you plenty of time to sharpen your skills.

Certification exam preparation

Get ready for your Citrix certification exam by practicing test materials covered by lab exercises.

Module 2: Application Firewall Profiles and Policies

Overview:

Company ABC has previously deployed a NetScaler to load balance their primary web applications. You have been asked to configure Application Firewall protections in front of the demonstration web applications: AFWeb and WebGoat.

This module demonstrates the initial configuration of the Application Firewall by creating the necessary profiles and policies for each of the test applications. The initial profile state is tested to confirm that violations will be blocked and then the profiles will be modified until a more detailed configuration can be performed.

After completing this lab module, you will be able to:

- Create Application Firewall profiles with the appropriate initial settings to suit the application requirements.
- Configure profiles with responses to display when violations occur.
- Configure and bind Application Firewall policies to the appropriate virtual servers.
- Configure NetScaler Insight Center for AppFlow and Security Insight reporting for the Application Firewall feature.

This module contains the following exercises using the NetScaler Configuration Utility GUI:

• Exercise 2-1: Create Profile and Policy for AFWeb	20 min
• Exercise 2-2: Create Profile and Policy for WebGoat	10 min
• Exercise 2-3: Enable Insight Security Reporting	5 min

Before you begin:

Estimated time to complete this lab module: 35 minutes

Exercise 2-1: Create Profile and Policy for AFWeb

In this exercise, you will create the initial Application Firewall profile settings for AFWeb. This exercise will introduce the initial Application Firewall profile configuration and explore the default settings. Once the Application Firewall feature is confirmed to successfully block traffic, the profile will be minimally relaxed to allow normal page navigation until a more thorough configuration can be completed in Module 4. Application Firewall events logged to syslog will also be reviewed as part of the configuration and troubleshooting process.

This exercise prepares the initial profile settings and view the initial protection behavior.

Requirements for this scenario:

- Create a new Application Firewall profile and policy for AFWeb. Policy will be applied to AFWeb only.
- Configure initial profile settings so that violations are directed to the AFWeb error page: /blocked.htm.
- Test the profile to confirm block behavior is occurring and then disable blocking on Start URLs to allow normal application operation.

In this exercise, you will perform the following tasks:

- Configure Application Firewall Profiles and Policies for AFWeb

Configure Application Firewall Profiles and Policies for AFWeb

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. (Use Chrome for NetScaler Configuration Utility connections.) Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
2.	Test the connection to AFWeb without Application Firewall enabled: • Open Firefox and browse to **http://afweb.training.lab/**. Verify the AFWeb Home page is successfully displayed. • Open a second tab and browse to **http://172.21.10.111/**. Verify the AFWeb Home page is successfully displayed. **NOTE**: Use Firefox to test applications, while using Chrome to access the NetScaler Configuration Utility and make configuration changes.

3.	Enable basic features on the NetScaler: • Navigate to **System > Settings**. • Click **Configure Basic Features**. • Verify the following features are already enabled: o SSL Offloading o Load Balancing o HTTP Compression o Content Switching • Enable (check) the following features: o Rewrite o Application Firewall Click **OK**.
4.	Enable advanced features on the NetScaler: • Navigate to **System > Settings**. • Click **Configure Advanced Features**. • Keep default features enabled (Surge Protection and Web Logging). • Enable (check) the following additional features: o Responder Click **OK**.
5.	Change global HTTP Cookie version: • Navigate to **System > Settings**. • Click **Change HTTP Parameters**. • Select **Version1** under Cookie. Click **OK**.
6.	Create an Application Firewall profile for AFWeb: • Navigate to **Security > Application Firewall > Profiles**. • Click **Add**. Create a profile for web applications with basic default settings: • Enter **appfw_prof_afweb** in the Name field. • Select **Web Application (HTML)** under Profile Type. • Select **Basic** under Defaults. Click **OK**.
7.	Update the profile with initial profile settings: • Select (check) profile **appfw_prof_afweb** and click **Edit**. • Click **Profile Settings** in the right pane (under Advanced Settings) to add the category to the configuration area. Configure the Redirect URL for AFWeb: • Select **Redirect URL** under HTML Error. • Enter **/blocked.htm** in the Redirect URL field. • Click **OK** to apply changes to Profile Settings.

8.	Verify Security Checks initial settings: • Click **Security Checks** under Advanced Settings. • Verify that all basic mode security checks are enabled for Block, Log, and Stats. Start URL Deny URL Buffer Overflow Field Formats HTML Cross-Site Scripting HTML SQL Injection • Verify that Learning is disabled for all security checks. • Click **OK**. Verify initial Start URL settings: • Click **Relaxation Rules** under Advanced Settings. • Select (check) **Start URL** and click **Edit**. • View the default Start URLs that are currently enabled. • Click **Close**. Click **Done** to close the profile properties.
9.	Create an Application Firewall policy for AFWeb: • Navigate to **Security > Application Firewall > Policies > Firewall**. • Click **Add**. • Enter **appfw_pol_afweb** in the Name field. • Select **appfw_prof_afweb** in the Profile drop-down list. • Enter the following in the Expression field. (Default policy engine true value constant.) `true` Click **Create**.
10.	Bind the Application Firewall policy to the load balancing virtual server for AFWeb. Select the bind point: • Click **Policy Manager** in the right-pane. • Select **Load Balancing Virtual Server** under Bind Point. • Select **lb_vsrv_afweb** under Virtual Server. • Click **Continue**. Bind the policy: • Click **Click to Select** under Select Policy. • Select **appfw_pol_afweb** and click **Select**. • Verify Priority is set to 100. • Click **Bind**. Click **Done**.
11.	Test the initial profile settings: • Open Firefox and browse to **http://afweb.training.lab/**. Verify the request is blocked by Application Firewall and you are seeing the /blocked.htm page instead. • Browse to **http://172.21.10.111/**. Verify the request is also blocked by Application Firewall. **NOTE**: Use Firefox to test applications, while using Chrome to access the NetScaler Configuration Utility and make configuration changes.

12. Connect to NS_VPX_01 using the NSIP Address (192.168.10.101) using the PuTTY SSH client.

- Use the PuTTY shortcut on the desktop of your Student Desktop OR run the following command: Right Click **Start > Run > Putty 192.168.10.101**

Log into the utility using the following credentials:

User Name: **nsroot**
Password: **nsroot**

13. Use the SSH session to view the Syslog events for AppFw:

Access Shell:

```
shell
```

Navigate to the Syslog directory:

```
cd /var/log/
```

View the current Syslog events for AppFw only:

```
more /var/log/ns.log | grep APPFW
```

Verify that syslog displays APPFW_STARTURL violations when attempting to access http://afweb.training.lab/ or http://172.21.10.111/.

Verify the action is identified as <blocked>.

14. View Syslog events for AppFw as they occur:

Run the following command to output APPFW events as they are generated:

```
tail -f /var/log/ns.log | grep APPFW
```

Keep the command running. This window will be referred to as **Putty (1)**.

15. Switch to Firefox and generate a new violation:

- Browse to **http://afweb.training.lab/** again.
- Browse to **http://172.21.10.111/** again.

16. Return to the **Putty (1)** window displaying the Syslog output.

- Verify new violations are displayed as they occur.

IMPORTANT: Keep this putty session with the log output running throughout the Application Firewall exercises. This window will be referred to as **Putty (1)** and will be used to view the current syslog events for the Application Firewall in later exercises.

If the output stops due to a log file rollover event, use the following procedure:

- Use **CTRL+C** to stop the current command.
- Then repeat the log output command:

```
tail -f ns.log | grep APPFW
```

Stop and restart output as needed.

17.	Update the Application Firewall profile settings for AFWeb to temporarily prevent blocking on Start URL violations: • Return to the NetScaler Configuration Utility in Chrome at http://192.168.10.101. • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_afweb** and click **Edit**. Update the Start URL Settings: • Select **Security Checks** under Advanced Settings. • Select (check) **Start URL** and click **Action Settings**. • Uncheck **Block**. • Click **OK** to apply the change to Start URL Settings. Click **OK** under Security Checks. Click **Done**. **NOTE**: When editing a profile the "Save & Close" option is presented to make it easy to apply all security check changes and close profile in one click; however, it does not save the NetScaler configuration. It is not a required step. If you click "OK" under the Security Check section, then all security check changes will be applied. If you click "DONE", the profile will be closed. The lab procedures will not use the "Save & Close" button as in some situations the profile will need to remain open for other edits. For consistency, the lab will require student to click "OK" under sections where settings have been changed. Then the click "Done" step will be used to close the profile when all edits are complete, when needed.
18.	Test the profile settings: • Open Firefox and browse to **http://afweb.training.lab/**. Verify the request is not blocked and the AFWeb Home is displayed. • Browse to **http://172.21.10.111**/. Verify the request is not blocked and the AFWeb Home is displayed. Note: Make sure that page /**blocked.htm** is not displayed in the URL when testing.
19.	Return to the **PuTTY (1)** SSH session displaying the Syslog output: • Verify additional APPFW_STARTURL violations are displayed, but this time the actions indicate the content was <not blocked>. **NOTE**: • Disabling the BLOCK action, does not disable the security check or prevent the processing associated with it. • The violation is still detected, only the NetScaler does not take a corrective action; the violation is still logged as the LOG action is enabled.
20.	Return to the GUI and save the NetScaler configuration.

Takeaways:

- Application Firewall profiles can be configured with default settings that determine the initial start state. The basic defaults start with basic protections, no sessionization-based features enabled, and no learning enabled. The advanced defaults start with advanced protections, including sessionization-based features and URL Closure enabled, no default Start URL rules, and learning enabled.

- The security checks included are determined by the profile type: Web, XML, or Web 2.0.
- Even though the basic profile contains an initial set of Start URL's, most profiles will require some minimal settings in order to successfully pass traffic.
- Each profile can be configured with a unique blocked page to display on violation either by redirecting to a specific error page, redirecting to the default page "/", or by importing content for the NetScaler to display on violation.

Exercise 2-2: Create Profile and Policy for WebGoat

In this exercise, you will create an additional Application Firewall profile for use with WebGoat. The initial profile settings will be enabled, the HTML Error page will be specified, and the default protection level should be tested to confirm that WebGoat traffic is being blocked successfully. Finally, the profile will be updated to prevent blocking based on the initial Start URL violations, until a more thorough configuration can be completed in Module 4.

The WebGoat profile will use a similar set of initial configurations as the AFWeb profile to start with. Each application will then require unique protection settings to ensure security protections while allowing successful application operation in later modules.

Requirements for this Scenario:

- Create a new Application Firewall profile and policy for WebGoat. Policy will be applied to WebGoat virtual server only.
- Create an HTML Error page on the NetScaler to use as a violations blocked page.
- Test the profile to confirm block behavior is occurring and then disable blocking on Start URLs to allow normal application operation.

In this exercise, you will perform the following tasks:

- Configure Application Firewall Profiles and Policies for WebGoat.

Configure Application Firewall Profiles and Policies for WebGoat

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. (Use Chrome for NetScaler Configuration Utility connections.) Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
2.	Test the connection to WebGoat without Application Firewall protection configured: • Open Firefox and browse to **http://webgoat.training.lab/WebGoat/attack**. **NOTE**: The WebGoat URL path is case-sensitive. • Log on as **guest** / **guest**. • Click **Start WebGoat**. • Verify you can see the WebGoat site page. **NOTE**: In future exercises, if the browser is reset or cookies are deleted, you may need to repeat this process. If prompted for authentication, use the **guest** credentials above. If you are presented with the "Start WebGoat" button, click "Start WebGoat" in order to proceed with the exercise, whether the exercise notes it or not.
3.	Return to the NetScaler Configuration utility in Chrome at http://192.168.10.101.

4.	Create an Application Firewall profile for WebGoat: - Navigate to **Security > Application Firewall > Profiles**. - Click **Add**. Create a profile for web applications with basic default settings: - Enter **appfw_prof_webgoat** in the Name field. - Select **Web Application (HTML)** under Profile Type. - Select **Basic** under Defaults. Click **OK**.
5.	Create a basic error page for use with Application Firewall violations for WebGoat: - Navigate to **Security > Application Firewall > Imports**. - Select **HTML Error Page** tab and click **Add**. - Enter **basicerror_webgoat** in the Name field. - Select **Text** under Import From. - Click **Continue**. - Enter the following text in the File Contents box: `This request was blocked by the Application Firewall.` - Click **Done**. **NOTE**: WebGoat does not have its own error page. The NetScaler is being used to provide a basic error message during Application Firewall violations. This content will be replaced with a more robust and user-friendly message later on in the course. The error message is being used for lab purposes to make it clear when the Application Firewall blocks an attack. Typically, in production, the "on error" response would be to redirect to "/" or to use a message that does not explicitly declare the use of a NetScaler Application Firewall in the protection process. Use care when wording a custom error message.
6.	Update the Application Firewall profile for WebGoat with initial settings: - Navigate to **Security > Application Firewall > Profiles**. - Select (check) profile **appfw_prof_webgoat** and click **Edit**. - Click **Profile Settings** in the right pane (under Advanced Settings) to add the category to the configuration area. Configure the Redirect URL for WebGoat: - Select **HTML Error Object** under HTML Error. - Select **basicerror_webgoat** in the HTLM Error Object drop-down list. - Click **OK** to apply changes to Profile Settings.

7.	Verify Security Checks initial settings: - Click **Security Checks** under Advanced Settings. - Verify that all basic mode security checks are enabled for Block, Log, and Stats. Start URL Deny URL Buffer Overflow Field Formats HTML Cross-Site Scripting HTML SQL Injection - Verify that Learning is disabled for all security checks. - Click **OK**. Verify initial Start URL settings: - Click **Relaxation Rules** under Advanced Settings. - Select **Start URL** and click **Edit**. - View the default Start URLs that are currently enabled. - Click **Close**. Click **Done**.
8.	Create an Application Firewall policy for WebGoat: - Navigate to **Security > Application Firewall > Policies > Firewall**. - Click **Add**. - Enter **appfw_pol_webgoat** in the Name field. - Select **appfw_prof_webgoat** in the Profile drop-down list. - Enter the following in the Expression field. (Default policy engine true value constant.) `true` Click **Create**.
9.	Bind the Application Firewall policy to the load balancing virtual server for WebGoat. Select the bind point: - Click **Policy Manager**. - Select **Load Balancing Virtual Server** under Bind Point. - Select **lb_vsrv_webgoat** under Virtual Server. - Click **Continue**. Bind the policy: - Click **Click to Select** under Select Policy. - Select **appfw_pol_webgoat** and click **Select**. - Verify Priority is set to 100. - Click **Bind**. Click **Done**.
10.	Test the initial profile settings: - Open Firefox and browse to **http://webgoat.training.lab/WebGoat/attack**. Verify the request is blocked by Application Firewall and you are seeing the imported error object generated by the NetScaler.

11.	Return to the **Putty (1)** session displaying the Syslog output: If necessary, restart the output by running: `tail -f ns.log | grep APPFW` Verify that syslog displays APPFW_STARTURL violations when attempting to access http://webgoat.training.lab/WebGoat/attack. Confirm the action taken indicates <blocked>.
12.	Update the Application Firewall profile settings for WebGoat to temporarily prevent blocking on Start URL violations: • Return to the NetScaler Configuration Utility in Chrome at http://192.168.10.101. • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_webgoat** and click **Edit**. Update the Start URL Settings: • Select **Security Checks** under Advanced Settings. • Select **Start URL** and click **Action Settings**. • Uncheck **Block**. • Click **OK** to apply the change to Start URL Settings. Click **OK** under Security Checks. Click **Done**.
13.	Test the updated profile settings: • Open Firefox and browse to **http://webgoat.training.lab/WebGoat/attack**. Verify the request is not blocked and the WebGoat Home page is displayed.
14.	Switch to the **Putty (1)** session running Syslog: • Verify additional APPFW_STARTURL violations are displayed, but this time the actions indicate the content was <not blocked>.
15.	Return to the GUI and save the NetScaler configuration.

Takeaways:

- Application Firewall Profiles can be tuned with application specific settings.
- AFWeb and WebGoat will have independent security settings, HTML Error/blocked page responses, and security check settings through application specific profiles.

Exercise 2-3: Enable Insight Security Reporting

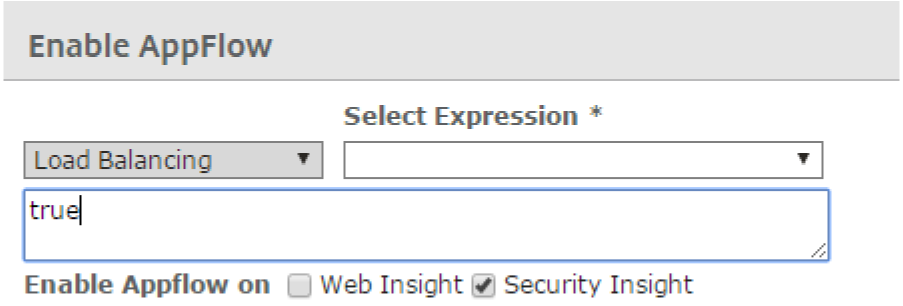
In this exercise, you will enable Insight Security reporting using NetScaler Insight Center. The NetScaler Insight Center will be used to apply the appropriate AppFlow configuration to the NetScaler for the AFWeb and WebGoat virtual servers. The Insight Security reporting will be enabled now to allow for later review of data gathering in the Application Firewall Logs and Troubleshooting module. A full discussion of Security Insight will be made after the Attacks and Protections exercises.

In this exercise, you will perform the following tasks:

- Integrate NetScaler Insight Center with the NS_VPX_01 NetScaler.
- Enable Security Insight reporting using Insight Center for the AFWeb and WebGoat load balancing virtual servers.
- View the configured AppFlow collector and AppFlow Policies and Actions configured by Insight Center.

Enable Insight Security Reporting

Step	Action
1.	Connect to the NetScaler Insight GUI. • Open a new tab **(Tab 2)** in Chrome and browse to **http://192.168.10.13**. Log in to Insight using the following credentials: User Name: **nsroot** Password: **nsroot** **NOTE**: About Insight Center • The default credentials for logging into Insight are nsroot / nsroot using either SSH or HTTP/HTTPS connections. • However, if attempting to log on to the local console (via the Hypervisor) the local credentials are root / nsroot. • We have not discussed the full functionality of Security Insight, however we perform configuration at this time to ensure we have collected data for the module later in class. •
2.	Configure NetScaler Insight Center: • Click **SKIP** to disable Citrix User Experience Improvement Program (CUXIP), if prompted. • Click **Get Started**. • Select **NetScaler**. • Enter **192.168.10.101** in the IP Address field. • Enter **nsroot** in the Username field. • Enter **nsroot** in the Password field. • Enter **nsroot** in the Confirm Password field. Click **Create**.

3.	Configure Insight Security monitor for the AFWeb and WebGoat virtual servers. • Select **Load Balancing** from the View field. • Select (check) each of the virtual servers: 172.21.10.111 lb_vsrv_afweb 172.21.10.112 lb_vsrv_webgoat • Click **Enable AppFlow**. Configure AppFlow Policy settings: • Enter **true** in the expression box (without using the Select Expression drop-down list). • Enable (check) **Security Insight**. Click **OK**.  Enable AppFlow Select Expression * Load Balancing true Enable Appflow on ☐ Web Insight ☑ Security Insight
4.	Verify both load balancing virtual servers display a green check under AppFlow Logging.
5.	Click **Dashboard** tab in NetScaler Insight Center. • Navigate to **Security Insight**. • Click **Get Started**. No data has been reported yet. This is expected.
6.	Switch to the NetScaler Configuration Utility for NS_VPX_01 (NSIP): • Connect to http://192.168.10.101. • Log on as nsroot / nsroot.

7.	View the AppFlow settings configured by NetScaler Insight Center: View Features: • Navigate to **System > Settings**. • Click **Configure Advanced Features**. • Verify AppFlow is now enabled. • Click **OK**. View AppFlow Collector: • Navigate to **System > AppFlow > Collectors**. • Verify an entry for AppFlow with IP Address 192.168.10.13 is listed. View the AppFlow policies: • Navigate to **System > AppFlow > Policies**. • Verify a policy was created for both AFWeb and WebGoat virtual servers with an expression of **true**. View the AppFlow policy actions: • Navigate to **System > AppFlow > Actions**. • Select (check) policy **af_action_lb_vsrv_afweb_192.168.10.13** and click **Edit**. • Verify Security Insight is enabled and the AppFlow collector destination is specified. • Click **Close**. **NOTE**: • Enabling AppFlow with Security Insight using NetScaler Insight Center makes a number of configuration changes to the managed NetScaler appliance. These changes include enabling the AppFlow feature, configuring the AppFlow collector, and creating and binding the necessary AppFlow policies. • However, Security Insight requires some additional settings to be enabled, that Insight Center does not apply on its own. These settings will be enabled in Module 5.
8.	Save the NetScaler Configuration. (The configuration should report the running configuration has not changed, indicating Insight Center saved the configuration when applying its updates.)

Takeaways:

- NetScaler Insight Center simplifies the configuration of AppFlow integration by remotely configuring the managed NetScaler with the necessary AppFlow collectors and AppFlow policies.
- Web Insight and HTML Injection policies enable client/server web site performance data gathering for HTTP/HTTPS load balancing and content switching virtual servers.
- HDX Insight policies enable HDX (ICA Proxy) network metrics for NetScaler Gateway (SSL VPN virtual servers).
- Security Insight allows AppFlow reports on Application Firewall violations for protected load balancing and content switching virtual servers.

Module 4: Application Firewall: Attacks and Protections

Overview:

This module demonstrates the configuration and use of Application Firewall profiles and related settings to protect against multiple web application security vulnerabilities and attack vectors.

The exercises in this module demonstrate multiple web application security attacks and how Application Firewall profiles, signatures, and security checks are used to prevent the attack by blocking the attack or using an alternate attack mitigation option when available.

After completing this lab module, you will be able to:

- Configure additional Start URL regular expressions to allow legitimate traffic to be permitted.
    - Understand the difference between Start URL without URL closure and with URL Closure behavior.
    - Understand how URL Closure affects the Start URL configuration.
- Create and configure signature protections for a profile and manage rule states within the signature configuration.
- Understand the various web attacks and configure appropriate security checks to prevent the attacks.
- Manage protection actions enforced for individual security checks (Block, Log, Stats).
- Configure different levels of protection when needed by changing from attack prevention actions (Block) to attack negation actions (X-Out, Remove, and Transform).
- Configure Learning within the profile and then evaluate and deploy learned rules.
- Evaluate the NetScaler syslog (/var/log/ns.log) for Application Firewall events and use the syslog to confirm or troubleshoot profile configurations.

This module contains the following exercises using the NetScaler Configuration Utility GUI:

Block 1:	Total:	50 min	
• Exercise 4-1: Start URLs and Deny URLs			35 min
• Exercise 4-2: Application Firewall Signatures			10 min
• Exercise 4-3: HTML Comment Stripping			5 min
Block 2:	Total:	45 min	
• Exercise 4-4: Buffer Overflow			5 min
• Exercise 4-5: SQL Injection			30 min
• Exercise 4-6: Cross Site Scripting			10 min
Block 3:	Total:	35 min	
• Exercise 4-7: Cookie Consistency Check			25 min
• Exercise 4-8: Form Field Consistency Check			10 min
Block 4:	Total:	55 min	
• Exercise 4-9: Credit Card and Safe Objects			10 min
• Exercise 4-10: Start URLs with URL Closure / Learning			15 min
• Exercise 4-11: CSRF Form Tagging / Referer Header Validation			30 min

Before you begin:

Estimated time to complete this lab module: 3 hours 15 minutes

Exercises in this module will be performed in blocks. Recommended times per block are identified above. Timings may vary.

Exercise 4-1: Start URLs and Deny URLs

In this exercise, you will enable and update the Start URL security checks so that both AFWeb and WebGoat will properly display without being prematurely blocked. Additional adjustments to Start and Deny URLs will be made to ensure that required content is displayed, while preventing access to disallowed content.

Scenario:

During the initial set up of the Application Firewall, you noticed that the default profiles were being blocked even though the default Start URLs were enabled. Therefore, the first step in configuring the Application Firewall profiles is to update the Start URL settings so that legitimate application content will not be blocked once the protection is re-enabled.

After the Start URLs are properly configured to allow normal application traffic to be displayed successfully, additional adjustments will be made to protect additional areas of the site that are exposed due to Web Server misconfigurations, backdoors, or other broken navigation links.

This exercise will demonstrate the basic risk posed by forceful browsing and the use of both Start URLs and Deny URLs to prevent the vulnerability.

Requirements for this scenario for AFWeb:

- Update the Start URLs to allow basic AFWeb navigation to succeed while still preventing access to normally blocked content. Additional content will be allowed through additional modifications of the Start URLs. Other content will be added to the deny list.
  - In these exercises, it is important that the regular expressions added are properly configured to only allow or deny the content expected.
  - Ensure the regular expressions are not overly broad or overly narrow in scope.
- Configuration 1: With the default Start URLs configuration, the Application Firewall profile is automatically blocking navigation to the primary URLs for AFWeb: http://afweb.training.lab/ and http://172.21.10.111/.
  - Determine why these URLs are blocked and update the Start URLs to allow this content.
  - Identify if there is any other content that is still being blocked that should be expected to be allowed.
  - If properly configured, the Allow Demo page will still be blocked and the Deny Demo page is still allowed. This will be corrected next.
- Configuration 2: Make additional adjustments to the allow and deny lists.
  - Update the Start URL configuration to allow the Allow Demo page.
  - Update the configuration to explicitly block the Deny Demo page and the hidden page (/private.htm).

Requirements for this scenario for WebGoat:

- Configuration 1: With the default Start URL configuration, the WebGoat profile is automatically blocking most page navigation with the WebGoat website.
  - Beginning with the default page navigation to http://webgoat.training.lab/WebGoat/attack, identify why the violation is occurring and implement an appropriate Start URL.
  - It is important that you avoid configuring an overly broad Start URL.
- Configuration 2: After the initial WebGoat page has been allowed, additional site navigation will be blocked. Identify why the navigational links are being blocked and identify a more appropriate allow Start URL.

In this exercise, you will perform the following tasks:

- Configure Start and Deny URLs for AFWeb
- Configure required Start URLs for WebGoat

Configure Start and Deny URLs for AFWeb

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. (Use Chrome.) Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
Configuration 1	
2.	Enable BLOCK action on Start URL: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_afweb** and click **Edit**. • Click **Security Checks** under Advanced Settings. • Check (enable) **Block** next to Start URL. • Click **OK** to apply changes to the Security Checks section. Remain in the profile settings; **do not** click Done.
3.	Clear all session state in Firefox before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate method: Click **History > Clear Recent History** to access manually. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared. • Re-open Firefox.
4.	Switch to Firefox and test the settings: • Browse to **http://afweb.training.lab/** Confirm the request is blocked by Application Firewall; the /blocked.htm page is displayed. • Browse to **http://172.21.10.111/** Confirm the request is blocked by Application Firewall. • Browse to **http://afweb.training.lab/index.htm** Confirm this request is allowed. **Challenge Question 1**: Why are the first two requests denied but the third request is allowed?____ (For more details see Question 1 in the Challenge Question & Answers section in the Appendix.)

5.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output. Identify which requests were blocked: • http://afweb.training.lab/ • http://172.21.10.111/ • http://afweb.training.lab/favicon.ico
6.	Return to the NetScaler Configuration Utility in Chrome.

7.	Update the Relaxation Rules for Start URLs:
	• Click **Relaxation Rules** under Advanced Settings.
	• Select (check) **Start URL** under Relaxation Rules and click **Edit**.
	Add the following Start URLs:
	• Click **Add**.
	• Enter the following regular expression in the Start URL field:
	`^http://afweb[.]training[.]lab/$`
	• Click **Create**.
	Repeat the procedure and add the following additional Start URLs:
	`^http://172\.21\.10\.111/$`
	`^[^?]+[.]ico$`
	Click **Close** to close the Start URL Relaxation Rules after all 3 rules have been created and return to the Profile properties view. (There will be 5 total rules in the Start URL list.)
	NOTE:
	• The second regular expression allows access to the URL via VIP. It is included for illustration purposes only.
	Challenge Question 2: Why was ^http://afweb[.]training[.]lab/$ better than ^http://afweb[.]training[.]lab/ for use as a Start URL: ______________________________________
	(For more details see Question 2 in the Challenge Question & Answers section in the Appendix.)

8.	Switch to Firefox and test the new settings: • Browse to **http://afweb.training.lab/** • Browse to **http://172.21.10.111/** • Browse to **http://afweb.training.lab/index.htm** Confirm all three URLs are now allowed and do not generate any blocked events.

Configuration 2	
9.	Attempt to navigate to the Allow Demo and Deny Demo pages. Use the browser back button to return to the main page between tests. • Click the **Allow Demo** link to navigate to the demonstration page. Confirm this page is currently blocked by Application Firewall. • Click the **Deny Demo** link to navigate to the demonstration page. Confirm this page is currently allowed by Application Firewall. • Manually browse to **http://afweb.training.lab/private.htm**. Confirm this page is currently allowed by Application Firewall. The objective with the next several steps is to update the Application Firewall profile so that: • The Allow Demo page is allowed. • The Deny Demo and the backdoor URL /private.htm are blocked. • You will update Start and Deny URLs to accomplish both tasks.
10.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output. Identify why the Allow Demo page was blocked: Start URL violation is identified for http://afweb.training.lab/allow.demo. Reason: URL is using a non-standard extension that is not accepted by the Start URLs and no additional Start URLs have been configured that explicitly allow the /allow.demo page.
11.	Return to the NetScaler Configuration Utility in Chrome. (Lab assumes the profile properties are still open.)
12.	Update the Relaxation Rules for Start URLs to allow the "Allow Demo" page: • Select (check) **Start URL** under Relaxation Rules and click **Edit**. • Click **Add**. • Enter the following regular expression in the Start URL field: **`/allow[.]demo$`** • Click **Create**. Click **Close** to close the Start URL Relaxation Rules summary.

13.	Update the Relaxation Rules for Deny URLs to block the "Deny Demo" page and the Private URL:
	• Deselect **Start URL** under Relaxation Rules.
	• Select (check) **Deny URL** under Relaxation Rules and click **Edit**.
	• Click **Add**.
	• Verify **Enabled** is checked to enable the rule (many Deny URLs are disabled by default so it is easy to inherit the wrong value.).
	• Enter the following regular expression in the Deny URL field:
	`/denyme[.]htm`
	• Click **Create**.
	
	Note: Make sure you have 50 per page selected at the bottom so you can see the new rules you are creating.
	
	Repeat the procedure and add the following additional Deny URL:
	• The Private URL:
	`/private[.]htm`
	• Click **Create**.
	
	Click **Close** to close the Deny URL Rules summary.

14.	Switch to Firefox and test the settings: • Browse to **http://afweb.training.lab/** Confirm the following are allowed: • Click the **Allow Demo** link. This should be successful. Confirm the following are blocked by Application Firewall: • Click the **Deny Demo** link. This should be blocked. • Manually browse to **http://afweb.training.lab/private.htm**. This should be blocked.
15.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify why the Allow Demo page was blocked: Deny URL violations are displayed for both /denyme.htm and /private.htm.
16.	Return to the NetScaler Configuration Utility in Chrome.
17.	Click **Done** to close the AFWeb profile.
18.	Save the NetScaler configuration.

Configure required Start URLs for WebGoat

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
Configuration 1	
2.	Enable BLOCK action on the Start URL: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_webgoat** and click **Edit**. • Click **Security Checks** under Advanced Settings. • Check (enable) **Block** next to Start URL. • Click **OK** to apply changes to the Security Checks section. Remain in the profile settings; do not click done.
3.	Disable the Application Firewall feature to test the unprotected WebGoat site. Disable feature using NetScaler CLI: • Open a second putty session (separate from the syslog session): Right Click **Start > Run > putty 192.168.10.101**. Log on as **nsroot** / **nsroot**. • Run the following command to disable Application Firewall: `disable ns feature appfw` • Keep this Putty session running for later exercises. This will be referred to as **Putty (2)**. **NOTE**: You will need to toggle the Application Firewall feature on and off throughout these exercises. The recommended procedure is to keep a CLI session running and run the appropriate enable/disable commands as needed. This will allow you to remain within the Profile configuration within the GUI for the subsequent steps.
4.	Switch to Firefox and browse to the WebGoat URL: **http://webgoat.training.lab/WebGoat/attack** Log on using the WebGoat credentials (if prompted): User Name: **guest** Password: **guest** If the "Start WebGoat" button is displayed, click **Start WebGoat**. **NOTE**: This step was performed to make sure WebGoat was operational before re-enabling the protection settings.
5.	Switch to **Putty (2)** and re-enable the Application Firewall feature using the NetScaler CLI: • Run the following command to enable Application Firewall: `enable ns feature appfw`

6.	Switch to Firefox and reconnect to the WebGoat URL: **http://webgoat.training.lab/WebGoat/attack** This time the request is blocked by Application Firewall. **Challenge Question 1** (for WebGoat): Why is this WebGoat URL blocked with the default profile?____ For more details see Question 1 in the Challenge Questions & Answers section in the Appendix.
7.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify why the WebGoat page (/WebGoat/attack) was blocked: Start URL violation identified for http://webgoat.training.lab/WebGoat/attack. URL does not conform to default Start URLs as it doesn't contain any extension. Violations may also be seen for /favicon.ico (if browser has been reset). During testing, you may or may not also see a violation for /favicon.ico. If the /favicon.ico is not displayed in the list of violations, it may be due to the browser caching this object when Application Firewall was disabled, this will be accounted for in later steps.
8.	Return to the NetScaler Configuration Utility in Chrome. (Lab assumes the profile properties are still open.)
9.	Update the Relaxation Rules for Start URLs to allow minimal required Access to WebGoat: • Click **Relaxation Rules** under Advanced Settings. • Select (check) **Start URL** under Relaxation Rules and click **Edit**. Add the following Start URLs: • Click **Add**. • Enter the following regular expression in the Start URL field: **`^http://webgoat[.]training[.]lab/WebGoat/attack$`** • Click **Create**. Click **Close** to close the Start URL Relaxation rules. Remain on the Profile Properties page. NOTE: Configuring Start URL rules to also allow access by VIP will be omitted for this application. All tests will be conducted using FQDN. However, remember to account for possible VIP access with production applications.
10.	Clear all session state in Firefox before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate method: Click **History > Clear Recent History**. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared. • Re-open Firefox. **Do not** browse to WebGoat yet.

11.	Switch to Firefox and reconnect to the WebGoat URL: **http://webgoat.training.lab/WebGoat/attack** If prompted, for credentials: • Log on as **guest / guest**. • Click **Start WebGoat** on the intro page. This time the request is allowed by Application Firewall.
Configuration 2	
12.	In WebGoat, navigate to the following links to test access. Return to the main WebGoat page between tests to resume navigation: • Click on **General > HTTP Basics** in the navigation pane on the left. • Click on **Access Control Flaws > Remote Admin Access** in the navigation pane on the left. These requests are blocked by Application Firewall. All lesson navigational links will be blocked.
13.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify which URLs are blocked and why, based on the violations listed in the syslog output. The navigational links in WebGoat use the form: • http://webgoat.training.lab/WebGoat/attack?Screen=###&menu=### **Challenge Question 3**: Why is the base URL created in the previous step along with the default Start URLs, not enough to allow access to WebGoat?____ For more details see, Question 3 in the Challenge Question & Answers in the Appendix.
14.	Return to the NetScaler Configuration Utility in Chrome.
15.	Update the Start URLs in the WebGoat profile to allow access to WebGoat content that references query parameters. • Select (check) **Start URL** under Relaxation Rules and click **Edit**. • Click **Add**. • Enter the following regular expression in the Start URL field: `^http://webgoat[.]training[.]lab/WebGoat/attack([?].*)?$` • Click **Create**. NOTE: This expression uses the query parameter regex that is included in the default Start URL and combines it with the WebGoat path. This will broadly allow any query parameter string concatenated with the /WebGoat/attack path. This may be overly broad for the legitimate URLs that WebGoat will be using, but we also need to prevent blocking URLs that conform to patterns for several of the attack demonstrations so that other attack/protection combinations can be used.

16. Add a Start URL for the favicon:
    - Click **Add**.
    - Enter the following regular expression in the Start URL field:
      ```
      /favicon[.]ico$
      ```
    - Click **Create**.

17. Disable the previous Start URL
    - Select Start URL rule **^http://webgoat[.]training[.]lab/WebGoat/attack$**
    - Click **Disable**. Click **Yes** to confirm.
    - Only 4 rules are enabled at this time: the two defaults, the new custom rule with the query parameters, and the /favicon.ico rule.

    Click **Close** to close the Start URL Relaxation Rules.

18. Clear all session state in Firefox before continuing:
    - Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**.
    - Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details.
    - Click **Clear Now**.

    IMPORTANT:
    - Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared.
    - Re-open Firefox. **Do not** browse to WebGoat yet.

19. Test the connection to WebGoat without Application Firewall protection configured:
    - Open Firefox and browse to **http://webgoat.training.lab/WebGoat/attack**.
      NOTE: The WebGoat URL path is case-sensitive.
    - Log on as **guest** / **guest**.
    - Click **Start WebGoat**.
    - Verify you can see the WebGoat site page.

    The request is allowed by Application Firewall.

    NOTE: In future exercises, if the browser is reset or cookies are deleted, you may need to repeat this process. If prompted for authentication, use the **guest** credentials above. If you are presented with the "Start WebGoat" button, click "Start WebGoat" in order to proceed with the exercise, whether the exercise notes it or not.

20. In WebGoat, navigate to the following links to test access:
    - Click on **General > HTTP Basics** in the navigation pane on the left.
    - Click on **Access Control Flaws > Remote Admin Access** in the navigation pane on the left.

    These request (and other navigational links) are now allowed by Application Firewall.

21. Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output:

    No new violations should be reported.

22. Return to the NetScaler Configuration Utility in Chrome.

23. Click **Done** to close the WebGoat profile.

24.	Save the NetScaler configuration. **IMPORTANT**: Complete this step before continuing.

Takeaways:

- When deploying a new Application Firewall profile, always start by configuring Start URLs to ensure successful access to a site.
- Start URLs (without URL Closure) provide a URL whitelist for site access. If a request is not covered by a Start URL, it will be blocked. In basic mode, the default Start URLs cover a wide range of content types, but even with these enabled, some sites may have URLs or content that does not conform to this initial pattern and additional Start URLs should be defined.
- Avoid creating unnecessarily broad Start URLs. Any content, not covered by a Start URL will be blocked anyway without the need to create an explicit Deny URL.

Exercise 4-2: Application Firewall Signatures

In this exercise, you will use Application Firewall signature protections as part of the Application Firewall profile. The signatures will be created from the default signatures within the NetScaler configuration and associated with the protection settings for WebGoat.

Scenario:

Now that the Application Firewall profiles have been initially configured to allow minimal required access to the applications, you have decided to add an extra layer of protection to WebGoat by incorporating the Signature protections for a hybrid security model.

During this initial round of configuration, you will only be enabling shell-shock protections as the security team wants to verify the protection behavior prior to rolling out other applications. Additional protections can be configured within the signature after the initial application evaluation phase.

NOTE: For lab purposes, in order to hack WebGoat for certain demonstrations in later exercises, several of the built-in signatures cannot be enabled. Please, only enable the settings indicated.

Requirements for this scenario:

- Create a custom signature from the built-in default signature list.
- Enable shell-shock signature protections and apply signatures to the WebGoat profile.
- Verify signature protections are in effect.

In this exercise, you will perform the following tasks:

- Configure Signature Protections for WebGoat.
- Identify signature violation events in syslog.

Configure Signature Protections for WebGoat

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
2.	Update signatures: • Navigate to **Security > Application Firewall > Signatures**. • Select (Check) ***Default Signatures** and then click **Update Version**. Version 13 was the signature version available with the NetScaler 11.1.47.14 firmware. Newer signature versions may be availbale. • If an update to signature version 14 (or later) is available, select **OK** to update the version. • Notice the Base version for signatures on the NetScaler is incremented to the new version if updated.

3.	Create a new custom signature based on the default signature file: • Select (Check) ***Default Signatures** and click **Add**. (This will create a new signature instance based on the current default signature definition.) Do not click on *Default Signatures or you will automatically open the default signatures to edit. Instead, check the option so that Add command will create a new signature instance based on the Default Signatures. • Enter **WebGoatSigs** in the Name field. • Click **Toggle All** under Signature Rules to disable the list of signatures to display. (Toggle does not enable or disable rules, but changes the signature categories to show or hide.) Keep the Signature properties open.
4.	Enable the Shell Shock protection rules: • Select (check) category **web-shell-shock** and click **Show/Hide** to display the rules in this category. • Verify the web-shell-shock rules are displayed only (4 rules). (Rule IDs 999976-999979) • Click **Action** in the right-pane and click **Enable all**. • Click **Yes** to confirm. Hide the list of Shell Shock rules: • Select (check) **web-shell-shock** under Category in the left pane and click **Show/Hide**. (Note: The rules are still configured.)
5.	Enable an individual rule from the web-misc protection category: • Select **web-misc** under Category and click **Show/Hide**. • Notice in the Category pane (left pane) that it displays that you are on Page 1/24 with 475 rules (for the web-misc category only.) Use the Search function to find specific rules within the current selected category: • Click **Search** to open the Search pane. • Change the drop-down list from Enabled to **LogString**. • Enter **passw** in the search field next to the drop-down list that now says LogString. • Click **Refine Search**. Select the individual rule to enable: • Select (check) the individual rule: **Rule ID: 1122**: WEB-MIS /etc/passwd. • Click **Action > Enable rules** to enable the selected rule(s) only. Click **OK** to apply changes to custom signature. The new signature file "webgoatsigs" is created.

6.	Add custom signature settings to the WebGoat Application Firewall profile: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) profile **appfw_prof_webgoat** and click **Edit**. • Click **Profile Settings** under Advanced Settings. • Scroll down to the **Common Settings** section. • Select **webgoatsigs** under **Bound Signatures** • Click **OK** to apply changes to the Profile Settings. Click **Done** to close the profile properties.
7.	Verify WebGoat is still accessible after applying the signatures. Switch to Firefox and reconnect to the WebGoat URL: **http://webgoat.training.lab/WebGoat/attack**
8.	Verify the signature can block a risky request: • In WebGoat, navigate to **General > Http Basics**. • Enter the following in the Enter your Name field: `() { :;}; echo vulnerable` **NOTE**: This isn't a full shell shock attack but it meets the formatting requirements of the vulnerability. Verify content is blocked by Application Firewall. **NOTE**: Custom signatures are stored in the following locations: • /var/download/custom/<signature name> • /var/download/<signature name>
9.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output. View signature violation in syslog: Syslog will display an APPFW_SIGNATURE_MATCH violation, the rule ID, and rule description.
10.	Return to the NetScaler Configuration Utility in Chrome.
11.	Save the NetScaler configuration.

Takeaways:

- Signatures evaluate against requests before the security checks are evaluated.
- The NetScaler contains a default set of signatures that can be used to create custom signature definitions tailored for specific applications or web platforms.
    - Signature protections are NOT enabled in the default settings. The default signature cannot be changed.
    - By creating custom signatures from the default set, administrators receive the default signature protections and can enable or disable individual rules.
- Custom signature files can also be used to customize the SQL Injection and XSS (Cross-Site Scripting) patterns.
    - By default, applications are protected by the built-in SQL Injection and XSS settings.
    - A custom signature file can be used to define custom SQL keyword, special characters, wild card, and SQL transformation rules.

    - A custom signature file can also be used to define custom XSS allowed attribute and tag dictionaries along with custom denied patterns.
- Custom signature files can also be imported using a native file format supported by NetScaler or generated from external vulnerability scanning systems.

Exercise 4-3: HTML Comment Stripping

In this exercise, you will demonstrate sensitive information leak vulnerabilities present in existing HTML comments and update the Application Firewall protection to prevent this vulnerability.

Scenario:

The security team is concerned that some of the application developers have over-documented some of the production application code and as a result sensitive information such as application shortcuts and backdoors appropriate to developers are available to the public users of the site. One such vulnerability, involving both test and admin credentials, was already identified in an existing application.

While the application developers have a new mandate to sanitize their code prior to publishing on production servers, the security team is concerned about the delay in time to implementation and the reality that the developers will miss some instances of sensitive information.

The security team wants the Application Firewall configuration to protect against this specific vulnerability in a way that poses the least risk to the application functionality.

Requirements for this Scenario:

- Use WebGoat and view level of vulnerability in source code on the Code Quality > Discover Clues in the HTML lesson.
- Update the WebGoat application firewall profile to prevent sensitive information leaks within the source code.

In this exercise, you will perform the following tasks:

- View vulnerability for Sensitive Information Leaks with HTML Comments
- Prevent Sensitive Information Leaks with HTML Comment Stripping

View Vulnerability for Sensitive Information Leaks with HTML Comments

Step	Action
1.	Switch to Firefox and navigate to http://webgoat.training.lab/WebGoat/attack • In WebGoat, navigate to **Code Quality > Discover Clues in the HTML**. Search the page source for HTML comments that contain credentials for this page: • Right-click the page area near "Sign In" and click **View Page Source**. • In the Source pane, enter **CTRL+F** to search for the phrase **FIXME** in the page source. Search twice more (for FIXME) until you find the HTML comment with the credentials. Take note of where in the page source these credentials are listed. Keep the window with the page source open for later reference.
2.	Return to WebGoat page and confirm the credentials work: • Enter **admin** in the User Name field. • Enter **adminpw** in the Password. • Click **Login**. Confirm the credentials successfully logged in to the WebGoat page.

3.	Click **Restart this Lesson** to reset the lesson state.

Prevent Sensitive Information Leaks with HTML Comment Stripping

Step	Action
1.	Return to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101.
2.	Update the WebGoat profile settings to include HTML comment stripping: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_webgoat** and click **Edit**. • Click **Profile Settings** under Advanced Settings. • Select **Exclude Script Tag** under Strip HTML Comments. • Click **OK** to apply the changes to the profile settings. Click **Done**.
3.	Switch to Firefox and navigate to http://webgoat.training.lab/WebGoat/attack • In WebGoat, navigate to **Code Quality > Discover Clues in the HTML**. • Refresh the page. Search the page source for HTML comments that contain credentials for this page: • Right-click the page area near "Sign In" and click **View Page Source**. • In the Source pane, enter **CTRL+F** to search for the phrase **FIXME** in the page source. • If necessary, compare the new page source with the previous page source window. Near the second instance of "Fixme" in the page source you will now see the HTML Comment tags next to the "Sign In" header <h1> element: <!-- -->, but no comment content. This instance previously contained the logon credentials.
4.	Save the NetScaler configuration.

Takeaways:

- HTML Comments can be a useful source of information for an attacker as it could contain information about how the application works, including navigational links, backdoor URLs, production or test accounts that are still active on the system. Many developers don't think to clean this content before posting production code.
- HTML Comment Stripping is not enabled by default.
- NetScaler Administrators should evaluate whether comment stripping is required and whether "strip all comments" or "exclude script tags" are appropriate given the functionality of the website.
- If sensitive information cannot be removed using comment stripping (due to client-side code dependencies not covered by excluding the script tag), then the burden falls on the application developers to clean output in code prior to posting.

Exercise 4-4: Buffer Overflow

In this exercise, you will demonstrate a buffer overflow attack and protection. After the initial demonstration, you will then adjust the protection level for a specific application.

Scenario:

The application developer and the security team have reviewed the AFWeb application. They are concerned about some known buffer overflow vulnerabilities within their application and the web server platform it runs on. They want the NetScaler to be configured to protect against large data objects being submitted.

Requirements for this scenario:

- Enable Application Firewall for AFWeb and confirm that the Buffer Overflow protection can prevent large content submitted via URLs.
- After the initial configuration, the application developer is concerned that the initial thresholds are too aggressive and are actually blocking legitimate site traffic. The app developer wants you to adjust the allowed threshold to accept content that matches the application's allowed length but blocks content above the threshold specified.

In this exercise, you will perform the following tasks:

- Buffer Overflow - Attack and Exploit Demonstration (AFWeb)
  - Demonstrate attack against unprotected site.
  - Enable protection and observe default behavior.
  - Adjust security protection thresholds to allow previously blocked content.

Buffer Overflow - Attack and Protection Demonstration (AFWeb)

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
2.	Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI: • Run the following command to disable Application Firewall: `disable ns feature appfw`

3.	Open Firefox and browse to AFWeb: - Browse to **http://afweb.training.lab/**. - Click **Buffer Overflow Demo** link to access the demonstration page. - Verify the page loaded successfully. - View the URL used by the link. The Buffer Overflow demonstration page includes a very long URL. In some cases, manually supplying a large URL, large Cookie, or large header to a web site could cause a buffer overflow exception to occur. The Application Firewall will be used to prevent large URLs, headers, or cookies from being submitted.
4.	Switch to **Putty (2)** and re-enable the Application Firewall feature using the NetScaler CLI: - Run the following command to enable Application Firewall: `enable ns feature appfw`
5.	Switch to Firefox and return to http://afweb.training.lab/: - Click **Buffer Overflow Demonstration** link. - Verify this time the page is blocked by Application Firewall.
6.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify why the Buffer Overflow Demo page was blocked: A Buffer Overflow violation was reported for the reference URL exceeding the allowed length. Note that the syslog indicates the URL that caused the violation and its length, along with the maximum allowed length.
7.	Return to the NetScaler Configuration Utility in Chrome.
8.	Configure a relaxation in the Application Firewall Profile to allow access to the Buffer Overflow Demo page by adjusting the Buffer Overflow thresholds: - Navigate to **Security > Application Firewall > Profiles**. - Select (check) **appfw_prof_afweb** and click **Edit**. - Click **Security Checks** under Advanced Settings. - Select (check) **Buffer Overflow** under Security Checks and click **Action Settings**. - Enter **1228** in the Maximum URL Length. - Click **OK** to apply the Buffer Overflow Settings. Click **Done** to close the profile properties.
9.	Switch to Firefox and browse to http://afweb.training.lab/ - Click **Buffer Overflow Demo** link to access the demonstration page. Verify the page loaded successfully. - Click **Buffer Overflow 2 Demo** link to access the alternate demonstration page. Verify this page is blocked. Application Firewall will now allow URLs up to 1228 characters long, but will block URLs longer than this new threshold. This allows the demonstration to be viewed as a legitimate URL. While the second demonstration page is still in violation.
10.	Save the NetScaler configuration

Takeaways:

- Buffer Overflow attacks involve sending large amounts of data using long URLs, large cookies, or headers whose values exceed expected lengths.
- Buffer Overflow protection is enabled by default with maximum defined thresholds for URL Length, Cookie Length, and Header Length.
- The Buffer Overflow protection can be adjusted to set thresholds (larger or smaller) for acceptable content for your web application, while blocking everything over the allowed maximum threshold.

Exercise 4-5: SQL Injection

In this exercise, you will implement a protection against an application with a backend vulnerable to database exploits using SQL Injection attacks.

Scenario:

The security team and the application team are concerned about two of their applications that frontend databases. Both of these applications predated development standards that required the applications to use parameterized input for SQL queries or to ensure the applications properly sanitized user input prior to executing against the database. Until the code review has been completed, the security team wants you to update the Application Firewall protections to prevent certain patterns of SQL Injection attacks without compromising the current application functionality.

For this scenario, both AFWeb and WebGoat will be configured.

Requirements for this Scenario for WebGoat:

- Demonstrate the SQL Injection attack and level of vulnerability against WebGoat using the page: Injection Flaws > String SQL Injection.
- Enable Application Firewall protection to prevent the attack using the block action.
- Adjust the level of protection and negate the attack using the transform action.

Requirements for this Scenario for AFWeb:

- View the AFWeb application and view the level of vulnerability with two of its pages: SQL Injection Demo and Form Field Demo pages.
- Create a relaxation to allow the drop-down list on the Form Field Demo pages to not generate a violation, while leaving other fields on the site protected using the Block action.

In this exercise, you will perform the following tasks:

- SQL Injection 1 - Attack Demonstration and Block Protection (WebGoat)
- SQL Injection 2 - Configure Transform Action (WebGoat)
- SQL Injection 3 - Configure Relaxation with Field Exemption (AFWeb)

SQL Injection 1 - Attack Demonstration and Block Protection (WebGoat)

Step	Action
1.	Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI: • Run the following command to disable Application Firewall: `disable ns feature appfw`
2.	In Firefox, browse to http://webgoat.training.lab/WebGoat/attack.
3.	Access SQL Injection lesson in WebGoat: • Navigate to **Injection Flaws > String SQL Injection**.

4.	Demonstrate a SQL Injection attack against WebGoat: • Enter the following string in the **Enter your last name** field: `Smith' OR '1'='1` • Click **Go**. Verify the page now displays a list of all users and user data from the table and not just information for Smith.
5.	Switch to **Putty (2)** and re-enable the Application Firewall feature using the NetScaler CLI: • Run the following command to enable Application Firewall: `enable ns feature appfw`
6.	Switch to WebGoat in Firefox and reset the SQL Injection lesson: • Navigate to **Injection Flaws > String SQL Injection**. • Click **Restart this Lesson** at the top of the page.
7.	Repeat the SQL Injection attack against WebGoat: • Enter the following string in the **last name** field: `Smith' OR '1'='1` • Click **Go**. Verify the request was blocked by the Application Firewall.
8.	Switch to **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm syslog identified a SQL Injection violation. Identify the field name and URL where the attack occurred. • Field Name: account_name • URL: http://webgoat.training.lab/WebGoat/attack?Screen=XXXX&menu=1100

SQL Injection 2 - Configure Transform Action (WebGoat)

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**

2.	Update the WebGoat profile to transform SQL Injection attacks instead of block transactions: - Navigate to **Security > Application Firewall > Profiles**. - Select (check) **appfw_prof_webgoat** and click **Edit**. - Click **Security Checks** under Advanced Settings. Update Security Check to transform instead of block on violation: - Select (check) **HTML SQL Injection** under Security Checks and click **Action Settings**. - Disable (uncheck) **Block**. - Enable (check) **Transform SQL special characters**. - Click **OK** to apply changes to SQL Injection Security Check. Click **Done** to close the profile properties.
3.	Switch to WebGoat in Firefox and reset the SQL Injection lesson: - Return to http://webgoat.training.lab/WebGoat/attack. - Navigate to **Injection Flaws > String SQL Injection**. - Click **Restart this Lesson** at the top of the page, if needed.
4.	Repeat the SQL Injection attack against WebGoat: - Enter the following string in the **Enter your last name** field: **`Smith' OR '1'='1`** - Click **Go**. - Keep the WebGoat app on this page, for the time being. Confirm this time the request is allowed; but the attack is negated and no user data is displayed. Observe how the transformation affected the user input (which is echoed in the last name field) after submission. - Verify the text in the last name field displays the transformed expression: **`Smith'' OR ''1''=''1`** - Verify the message under the field, outputs the attempted SQL query with the transformed text include: **`SELECT * FROM user_data WHERE last_name = 'Smith'' OR ''1''=''1'`**
5.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm syslog identified a SQL Injection violation. However, this time the action is identified as <not blocked> and <transformed> in two successive events.

6.	Return to WebGoat in Firefox: • Return to the String SQL Injection page (Injection Flaws > String SQL Injection). • Click **Go** multiple times to resubmit the transformed content. Notice how the double single quotes ('') do not keep getting re-transformed after the initial request. The NetScaler's built-in transformation rules will transform a single-quote (') to a double single-quote (''), but a double single quote ('') is transformed to itself (''). This may or may not be appropriate for the application. **NOTE**: Transformations can be useful to negate an attack but may pose issues to an application on the backend if it is performing its own transformations or not expecting transformed characters.
7.	Return to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101.
8.	Update the WebGoat profile to restore the block action instead of the transform action: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_webgoat** and click **Edit**. • Click **Security Checks** under Advanced Settings. Restore Security Check to block on violation: • Select (check) **HTML SQL Injection** under Security Checks and click **Action Settings**. • Select (check) **Block** to enable it. • Deselect (uncheck) **Transform SQL special characters** to disable it. • Click **OK** to close the SQL Injection Security Check. Click **Done** to close the profile properties.

SQL Injection 3 - Configure Relaxation with Field Exemption (AFWeb)

Step	Action
1.	Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI: • Run the following command to disable Application Firewall: `disable ns feature appfw`

2.	Identify Field name and URLs used in the SQL Injection Demo page: Switch to Firefox and browse to AFWeb: - Browse to **http://afweb.training.lab**. - Click **SQL Injection Demo**. Identify field name: - In Firefox, Click **Tools > Web Developer Extension > Forms > Display Form Details**. - Identify the field name associated with the "Lookup Value" field: **search**. - Identify the URL where the field is located: **/sql.htm**. - Enter the following into the Lookup Value field: `Select '` - Click **Submit**. - Identify which URL the page is submitted to: **/sql.asp**. **NOTE**: This is one method for identifying the field names and their URLs. The field name can also be identified by viewing the page source, from the syslog entries when a violation is observed, or from the learned violations list once learning is enabled, in later exercises.
3.	Identify Field and URLs used in the Form Field Demo page: Return to AFWeb main page: - Browse to **http://afweb.training.lab**. - Click on **Form Field Demo** page. Identify field name: - View the options in the account type drop-down list. Notice that one of the options contains text that conforms to a SQL Injection attack: **President's Select Checking**. - In Firefox, Click **Tools > Web Developer Extension > Forms > Display Form Details**. - Identify the field name associated with the "Account Type" field: **acct_type**. - Identify the URL where the field is located: **/field.htm**. - Select the following option in the account type drop down list: **President's Select Checking**. - Click **Submit**. - Identify which URL the page is submitted to: **/field.asp**.
4.	Switch to **Putty (2)** and enable the Application Firewall feature in the CLI: - Run the following command to enable Application Firewall: `enable ns feature appfw`
5.	Return to Firefox and access AFWeb: - Browse to http://afweb.training.lab
6.	Perform the attack on the SQL Injection page: - Click **SQL Injection Demo**. - Enter the following into the Lookup Value field: `Select '` - Click **Submit**. Verify the attack was blocked.

7.	Perform the attack on the Form Field Demo page: • Return to http://afweb.training.lab. • Click **Form Field Demo**. • Select **President's Select Checking** in the account type field. • Click **Submit**. This submits the page to /field.asp. Verify the attack was blocked. The objective with the next several steps is to update the Application Firewall profile so that: • The search field on the SQL Injection Demo pages (/sql.htm and /sql.asp) is still protected from SQL Injection attacks. • While exempting the acct_type field on the Form Field Demo pages (/field.htm and /field.asp). Other fields on these pages will still be protected.
8.	Return to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101.
9.	Configure Relaxation Rule for HTML SQL Injection: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_afweb** and click **Edit**. • Click **Relaxation Rules** under Advanced Settings. • Select (check) **HTML SQL Injection** and click **Edit**. Exempt the SQL Injection field: • Click **Add**. • Enter **acct_type** in the Name field. • Enter the following regular expression in the URL field: `/field[.](htm|asp)` • Verify location is set to FORMFIELD. Click **Create** and click **Close** to apply the HTML SQL Injection Relaxation Rules. Click **Done** to close the profile.
10.	Save the NetScaler configuration.
11.	Return to Firefox and access AFWeb: • Browse to http://afweb.training.lab
12.	Perform the attack on the SQL Injection page: • Click **SQL Injection Demo**. • Enter the following into the Lookup Value field: `Select '` • Click **Submit**. Verify the attack was still blocked.

13.	Attempt to submit legitimate data on the Form Field Demo page: - Return to http://afweb.training.lab. - Click **Form Field Demo**. - Select **President's Select Checking** in the account type field. - Click **Submit** this submits the page to /field.asp. - Select **President's Select Checking** in the account type field again on /field.asp. - Click **Submit**. Verify all requests were allowed.
14.	Perform the attack on the protected field in Form Field Demo page: - Return to http://afweb.training.lab. - Click **Form Field Demo**. - Enter the following in the email address field: `Select '` - Click **Submit**. Verify attacks in the email_addr field are still blocked by Application Firewall.

Takeaways:

- The HTML SQL Injection security check can protect against SQL injection attacks against form fields, headers, and cookies within a web application.
- SQL Injection security check supports block on violation and transform on violation protections. The security check also supports learning for identifying relaxations (exemptions).
- Administrators can adjust the strictness of the SQL Injection security check per profile.
  - By default, a violation is only identify if both SQL Special Characters and SQL Keywords are present.
  - The strictness can be adjusted to violation on SQL Special Characters only, SQL Keywords only, Characters AND Keywords (default), and Characters OR Keywords.
  - By default SQL Wildcard characters do not trigger violations, but this can be enabled.
- If no signatures are bound to the profile, then the identification of SQL Characters and SQL Keywords, and the transformation rules are handled by the default behavior of the NetScaler. The default settings can be observed in the default signature.
  - If necessary, custom signatures can be created and bound to a profile that adjust the SQL Keyword, SQL Character, and SQL Wildcard dictionaries.
  - Custom signatures can also be used to customize the transformation rules.
  - Care should be exercised when changing from the default protections.

Exercise 4-6: Cross Site Scripting

In this exercise, you will demonstrate a Cross Site Scripting attack and implement protections against the attack.

Scenario:

The security team has alerted you to another potential issue with the WebGoat web application. The security team is concerned about Cross-Site Scripting attacks and wants to see a demonstration protecting against attacks uploaded to the application's built-in message system functionality. They will then consider implementing Cross-Site Scripting protections for other applications.

Requirements for this scenario:

- First, demonstrate the attack and vulnerability against an unprotected website using WebGoat's Cross-Site Scripting (XSS) > Stored XSS Attacks page.
- Next, enable the protection and confirm a cross-site scripting attack is blocked on violation.
- Then, change the application firewall protection to transform on violation.

In this exercise, you will perform the following tasks:

- Cross Site Scripting 1 - Attack Demonstration
- Cross Site Scripting 2 - Block Protection

Cross Site Scripting 1- Attack Demonstration

Step	Action
1.	Use **Putty (2)** to disable the Application Firewall feature using the NetScaler CLI: • Run the following command to disable Application Firewall: `disable ns feature appfw`
2.	In Firefox, browse to http://webgoat.training.lab/WebGoat/attack.
3.	Access Cross Site Scripting lesson in WebGoat: • Navigate to **Cross-Site Scripting (XSS) > Stored XSS Attacks**.
4.	Demonstrate a successful Cross Site Scripting attack: • Enter **badscript1** in the Title field. • Enter the following code in the Message field: `<script type="text/javascript">alert("Script Executed")</script>` • Click **Submit**. Verify the Message "badscript1" appears in the Message list. • Click **badscript1**. Verify the page executes the script. • Click **OK** to close the alert box. Note that if you navigate away from the page and then return, the message with the uploaded script is still present, meaning the attack will impact potentially other users of the site. Click **Restart this Lesson** before proceeding. (The message badscript1 will not be removed.)

Cross Site Scripting 2 - Block Protection

Step	Action
1.	Use **Putty (2)** to re-enable the Application Firewall feature using the NetScaler CLI: • Run the following command to enable Application Firewall: `enable ns feature appfw`
2.	Switch to the NetScaler Configuration Utility for NS_VPX_01 (NSIP): • Connect to http://192.168.10.101. • Log on as nsroot / nsroot.
3.	Open the WebGoat profile: • Navigate to **Security** > **Application Firewall** > **Profiles**. • Select (check) **appfw-prof_webgoat** and click **Edit**.
4.	View Security Check settings: • Click **Security Checks** under Advanced Settings. • Verify HTML Cross-Site Scripting is enabled for Block, Log, and Stats actions.
5.	Switch to Firefox and browse to WebGoat: http://webgoat.training.lab/WebGoat/attack • Navigate to **Cross-Site Scripting (XSS) > Stored XSS Attacks**.
6.	Repeat the Cross-Site Scripting attack: • Enter **badscript2** in the Title field. • Enter the following code in the Message field: `<script type="text/javascript">alert("Script Executed")</script>` • Click **Submit**. Confirm this attempt was blocked by the Application Firewall.
7.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm syslog identified a Cross Site Script (XSS) violation. Note the field name and URL where the attack occurred.
8.	Switch to Firefox and browse to WebGoat: http://webgoat.training.lab/WebGoat/attack • Navigate to **Cross-Site Scripting (XSS) > Stored XSS Attacks**. Verify the previous message badscript1 is displayed in the message list, but badscript2 was prevented and is not present.
9.	Return to the NetScaler Configuration Utility in Chrome.

10. Update the WebGoat profile to transform Cross Site Scripting attacks instead of block transactions.

- Navigate to **Security > Application Firewall > Profiles**.
- Select (check) **appfw_prof_webgoat** and click **Edit**.
- Click **Security Checks** under Advanced Settings.

Configure transform action for Cross-Site Scripting:

- Select (check) **HTML Cross-Site Scripting** under Security Checks and click **Action Settings**.
- Disable (uncheck) **Block**.
- Enable (check) **Transform cross-site scripts**.
- Click **OK** to apply changes to the Cross-Site Scripting Security Check.

Click **Done** to close the profile settings.

11. Save the NetScaler Configuration.

12. Switch to Firefox and browse to WebGoat: http://webgoat.training.lab/WebGoat/attack

- Navigate to **Cross-Site Scripting (XSS) > Stored XSS Attacks**.

13. Repeat the Cross-Site Scripting attack:

- Enter **badscript3** in the Title field.
- Enter the following code in the Message field:

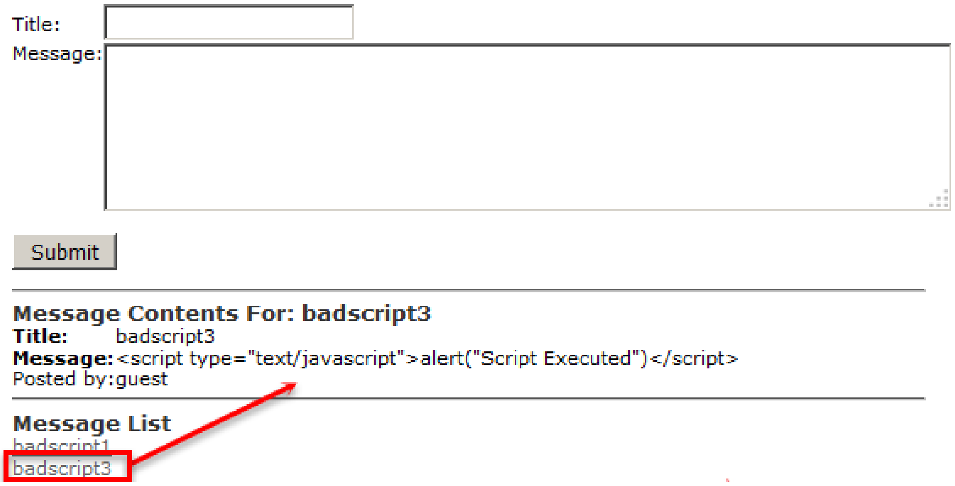
```
<script type="text/javascript">alert("Script
Executed")</script>
```

- Click **Submit**.

Confirm this attempt was allowed, but transformed by the Application Firewall.

- Click **badscript3** in the Message List.
- Note that the script content is displayed in the Message body, but the script is not executed (like it was in badscript1).

The message contents were safely transformed, so the message posts but as non-executable code. Message body is displayed instead of an alert prompt.



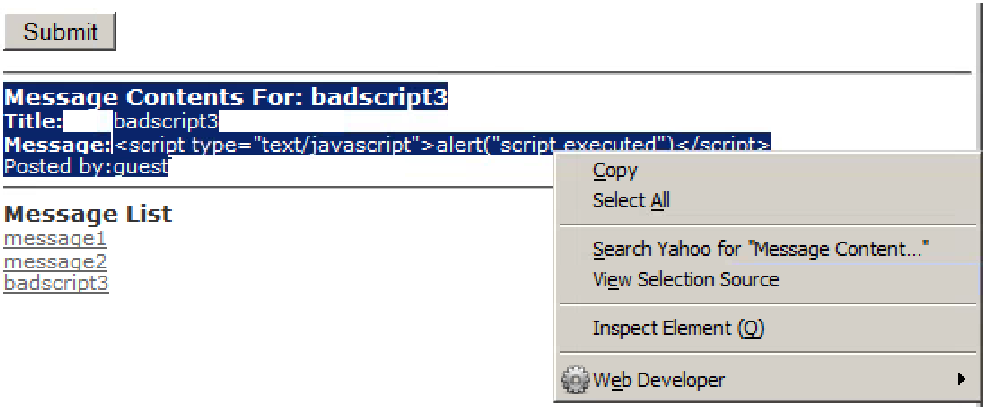
Message Contents For: badscript3
Title: badscript3
Message: <script type="text/javascript">alert("Script Executed")</script>
Posted by:guest

Message List
badscript1
badscript3

14.

In Firefox, highlight the Message Contents for badscript3, right-click and click **View Selection Source**. This will display the page source for the highlighted section only.

Highlight This:

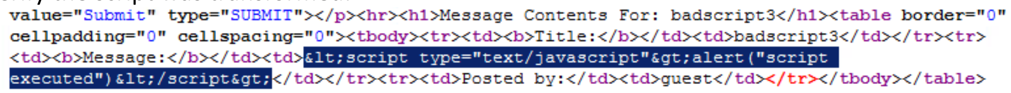


Submit

Message Contents For: badscript3
Title: badscript3
Message: <script type="text/javascript">alert("script executed")</script>
Posted by:guest

Message List
message1
message2
badscript3

Verify the script was transformed:



```
value="Submit" type="SUBMIT"></p><hr><h1>Message Contents For: badscript3</h1><table border="0"
cellpadding="0" cellspacing="0"><tbody><tr><td><b>Title:</b></td><td>badscript3</td></tr><tr>
<td><b>Message:</b></td><td>&lt;script type="text/javascript"&gt;alert("script
executed")&lt;/script&gt;</td></tr><tr><td>Posted by:</td><td>guest</td></tr></tbody></table>
```

Transformations:

- "<" was transformed to &lt;
- ">" was transformed to &gt;

15.

Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output:

Identify the violation displayed:
Confirm syslog identified a Cross Site Script (XSS) violation. The bad tag "script" was not blocked. While the cross-site script special characters were transformed.

Takeaways:

- The Cross Site Scripting security check can protect against XSS injection attacks against fields, headers, and cookies within an application. The Cross-Site Scripting can target tags, attributes, and patterns.
- Cross Site Scripting security check supports block on violation or transform on violation protections. The security check also supports learning.
- Violations of the security check include:
    - Presence of tags or attributes not listed in the allowed tags or allowed attributes lists.
    - Presence of content that confirms to a pattern in the denied patterns list.
- Cross-Site Script security check settings are based on the settings specified in the default signatures. An administrator can customize the allowed/denied settings by creating a custom signature file, adjusting the XSS patterns, and then binding the custom signature to the appropriate application.
- In addition, Cross-Site Scripting enforces a single origin-rule, meaning scripts should only access or modify content on the server on which they are located. Many applications may use extensive libraries of JavaScript-enhanced web content that violates this rule. The Cross-Site Security check behavior must be adjusted.

- Upon transformation, the NetScaler encodes the "<" and ">" script tag delineators with their html encoded counterparts: &lt; or &gt;.
- NetScaler 11.0 and later implements a change to the violation triggers for Cross-Site Scripting that is different than prior releases.
    - Previously, the presence of unmatched brackets could trigger a violation. Example a "<" without a corresponding ">", or two empty brackets with no content between "<>".
    - To support streaming content, where unmatched brackets can occur, cross-site script now only triggers if both tags occur "<" followed by a ">".

Exercise 4-7: Cookie Consistency Check

In this exercise, you will demonstrate a cookie tampering attack and configure the NetScaler Application Firewall to prevent the attack.

Scenario:

The security team wants to see a demonstration of how to use the Application Firewall to prevent cookie tampering and poisoning attacks. For this example, AFWeb will be used since it creates both persistent and transient cookies as part of the application operation.

The security team first wants to see what the attack prevention behavior is like using the block action. The team also wants to better understand the tracking process the NetScaler users for the cookies generated by the application, in order to ensure the various application development teams that the NetScaler protection should not "break the application". Prior to performing an attack, the security team wants to be sure you demonstrate normal application operation with the protections in place before performing the attack demonstration.

After the basic demonstration, the security team has heard some troubling reports from some of their application developers that a few of the legacy applications are storing sensitive information in cookies used to process user logon or application security decisions. These particular applications are slated to be rewritten to address this concern, but you have offered to show the Cookie Consistency check transformation action as well, in order to mitigate some of these risks. Applications will need to be tested to ensure that the transformation actions do not create an unexpected issue for the necessary applications.

Requirements for this scenario:

- Demonstrate the cookie tampering attack using AFWeb with protections disabled.
    - Use the AFWeb Cookie Consistency Demo page and view the normal page behavior while unprotected.
    - Use browser plug-ins to view cookie state and properties as the cookies are created, modified, and deleted.
- Next, enable the Cookie Consistency Check with Block action and repeat the attack while the protection features are enabled. Observe the protection behavior and violation events.
- Then, update the Cookie Consistency Check to protect cookies with the Transformation settings and repeat the attack while investigating the transformation options.
    - The transformation lab will demonstrate cookie transformation by encrypting cookies, proxying session cookies, and modifying cookie HTTPOnly and Secure flags.
- Finally, update the profile with a relaxation to create an exemption for the cookie.

In this exercise, you will perform the following tasks:

- Cookie Consistency 1 - View Cookies and Perform Cookie Tampering Exploit
- Cookie Consistency 2 - Attack Protection with Block Action
- Cookie Consistency 3 - Cookie Transformation Protection
- Cookie Consistency 4 - Configure Relaxation

Cookie Consistency 1- View Cookies and Perform Cookie Tampering Exploit

Step	Action
1.	Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI: • Run the following command to disable Application Firewall: `disable ns feature appfw`
2.	Clear Session state in Firefox before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate: Click **History > Clear Recent History**. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared. **IMPORTANT**: • For this exercise, make NetScaler configuration changes in Chrome and test AFWeb in a single instance of Firefox. • During this exercise, you will be asked to clear session state and cookies in Firefox multiple times. Be sure you close and restart all instances of Firefox when asked. • When viewing output in LiveHTTPHeaders, you may capture some background process from Firefox. Ignore this output. Most of these services were disabled, but a few still occur.
3.	In Firefox, browse to AFWeb: http://afweb.training.lab: • Open a new instance of Firefox and browse to the main page: http://afweb.training.lab. • For this exercise, you must connect consistently using the FQDN. (If you switch between FQDN and IP Address, you will have inconsistent results.) Open Live HTTP Headers in a new tab: • Click **Tools > Live HTTP Headers**. • AFWeb should be in one tab; with Live HTTP Headers in another. Note: There are 2 tools menu's select the one at the top next to bookmarks. Return to the AFWeb tab: • Refresh the main page. Switch to Live HTTP Headers tab: • Scroll to the bottom of the output and verify in the final request no Cookie headers are present.

4.	Access the Cookie Consistency Demonstration Page: Return to the AFWeb tab: • Click **Cookie Consistency Demo** to navigate to /cookie.asp. Switch to the Live HTTP Headers tab: • Confirm in the final request the ASPSESSIONID cookie is created. Return to the AFWeb tab:
5.	Use the Cookie Demonstration page to set a cookie • Enter **<your name>** in the Name field. • Click **Submit**. Verify the Cookie Demonstration page, now displays the following: • <Your Name> is set as the value of the cookie and is displayed on the page. • The page gives you options to Delete or Modify the cookie. Take no action. Switch to Live HTTP Headers to view the cookie(s) that are set: • The final request now lists multiple cookies: ASPSESSIONID (there will be at least one cookie present) MySiteVisitorName Web Developer Extension can also be used to view Cookies (and Cookie parameters) in Firefox: • Switch to AFWeb tab. • Click **Tools > Web Developer Extension > Cookies > View Cookie Information**. • The current cookie details will then be displayed in a new tab. Use either Live HTTP Headers or Web Developer Extension to identify details such as: • Which cookies are temporary or persistent? • Are cookie contents encrypted or clear text? • Are cookie flags for HTTPOnly or Secure set or not set?

6.	Continue with the Cookie tampering exercise using the Cookie Consistency Demo page: Return to the AFWeb tab: • Click **Modify** to change the value of your cookie from <your name> to the new value "Modified by Client". • Once the cookie has been modified the new value will be presented. • If you navigate to http://afweb.training.lab/allow.demo and then back to http://afweb.training.lab/cookie.asp, the modified value will still be present. • If you close (without clearing cookies) and open the browser, the modified cookie will persist. Remember to start LiveHttpHeaders again. Use Web Developer Extension to view new cookie details: • Click **Tools > Web Developer Extension > Cookies > View Cookie Information**. • This will open a new tab with the current cookie information. **NOTE**: The "modify" button uses client side code to change the cookie without having the server perform the modification. Other utilities such as Tamper Data could have been used to manipulate this cookie as well. This type of client-side manipulation of the cookie is what the NetScaler will be used to prevent.
7.	Delete the MySiteVisitorName cookie: Return to the AFWeb tab: • Click **Delete** to delete the MySiteVisitorName cookie. This resets the page and returns you to the initial state, prompting you to enter your name. However, the ASP Session cookie is still set. Switch to Live HTTP Headers to view the session cookie that remains: • ASPSESSIONID is still present. • MySiteVisitorName is deleted. **NOTE**: The Cookie Demonstration page has three states: • No MySiteVisitorName cookie set; therefore the page prompts for your name. • MySiteVisitorName cookie set with your custom value. The page now displays your custom value and gives you the option to Modify or Delete the cookie. • MySiteVisitorName cookie set with predefined "modified" value. The page now displays the value "Modified by Client" instead of your custom value and includes options to modify or delete the cookie. The page state displayed is directly related to the state of the MySiteVisitorName cookie. Be aware of this as we use the NetScaler to prevent the cookie tampering attack.

8.	Clear all cookies to fully reset demonstration: - Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate: Click **History > Clear Recent History**. - Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. - Click **Clear Now**. **IMPORTANT**: - Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared.

Cookie Consistency 2 - Attack Protection with Block Action

Step	Action
1.	Switch to **Putty (2)** and re-enable the Application Firewall feature using the NetScaler CLI: - Run the following command to enable Application Firewall: `enable ns feature appfw` **NOTE**: Cookie Consistency check is currently disabled as it is not on by default with the Basic settings.
2.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
3.	Enable the Cookie Consistency security check: - Navigate to **Security > Application Firewall > Profiles**. - Select (check) **appfw_prof_afweb** and click **Edit**. - Click **Security Checks** under Advanced Settings. - Select (check) **Cookie Consistency** under Security Checks and click **Action Settings**. - Enable (check) **Block**. - Enable (check) **Log**. - Enable (check) **Stats**. Click **OK** to apply Cookie Consistency settings. Click **Save and Close** under Security Checks to apply settings to current profile.
4.	Open Firefox and access the AFWeb site: - Browse to **http://afweb.training.lab/**. - **Do not** access the cookie demonstration page yet. Use Web Developer Extension to view the current cookies on the site: - Click **Tools > Web Developer Extension > Cookies > View Cookie Information**. - Switch to the Cookie Information tab and confirm one cookie exists: citrix_ns_id This is the NetScaler sessionization tracking cookie.

5.	Next, access the Cookie Demonstration page: - Switch to the AFWeb tab. - Click **Cookie Consistency Demo** link. Use Web Developer Extension to view the current cookies on the site: - Click **Tools > Web Developer Extension > Cookies > View Cookie Information** again. - Switch to the second Cookie Information tab and confirm three cookies exist: citrix_ns_id This is the NetScaler sessionization tracking cookie. citrix_ns_id_.training.lab_%2F_wat Tracking cookie for other session cookies. ASPSESSIONID Session cookie created by the AFWeb application.
6.	Next, set the cookie with the custom value (This is not an attack): - Switch to the AFWeb tab. - Enter **<your name>** in the Name field. - Click **Submit**. Verify the Cookie Demonstration page now displays the value of your cookie: <your name>. Use Web Developer Extension to view the current cookies on the site now: - Click **Tools > Web Developer Extension > Cookies > View Cookie Information** again. - Switch to the third Cookie Information tab and confirm five (5) cookies exist: citrix_ns_id This is the NetScaler sessionization tracking cookie. citrix_ns_id_.training.lab_%2F_wat Tracking cookie for other session cookies. **citrix_ns_id_.training.lab_%2F_wlf** Tracking cookie for other persistent cookies. **MySiteVisitorName** Persistent cookie created by the AFWeb application. ASPSESSIONID Session cookie created by the AFWeb application. **NOTE**: These lab steps were included to demonstrate how the Cookie Consistency check tracks sessionization and the cookies created by the application before showing the attack negation.
7.	Switch to the AFWeb tab: - Click **Modify** to modify the cookie from <your name> to "Modified by Client". Verify that you are taken to the "What is your name?" prompt which indicates no cookie is set instead of to the "Modified by Client" page. **NOTE**: With the Cookie Consistency Check, on violation the BLOCK action does not redirect the user to the specified HTML error page, instead the illegally modified cookie is stripped from the request by the NetScaler and never reaches the Web Server. The cookie will still be in the client side page, but it is not reaching the web server during the request. As a result, there will still be 5 cookies present in the browser when viewed using Web Developer Extension.

8.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm a Cookie Consistency check violation was identified for the MySiteVisitorName cookie. Syslog indicates the action taken was <blocked>. Also note the cookie is in violation from multiple URLs as the cookie is presented in multiple requests for the different page objects (.css, images, etc.).
9.	Clear all cookies to fully reset demonstration before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate: Click **History > Clear Recent History**. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared.

Cookie Consistency 3 - Cookie Transformation Protection

Step	Action
1.	Return to the NetScaler Configuration utility and access the AFWeb Profile. • Navigate to **Security > Application Firewall > Profiles**. • Select **appfw_prof_afweb** and click **Edit**.
2.	Change the Cookie Consistency Check behavior from Block to Transform: • Click **Security Checks** under Advanced Settings. • Select (check) **Cookie Consistency** under Security Checks and click **Action Settings**. • Enable (check) **Block**. (Keep enabled.) • Enable (check) **Transform**. Configure the following Transformation settings: • Encrypt Server Cookies: **Encrypt All**. • Proxy Server Cookies: **Session Only**. • Flags to Add In Cookies: **All**. Click **OK** to apply Cookie Consistency Settings. Click **Done**. **NOTE**: The Cookie Consistency check is unique in that the BLOCK and TRANSFORM actions can be enabled at the same time. Cookie Transformation changes the way Cookie Consistency checks work. Enabling Transform actions only without a Block action may not protect against all attacks.

3.	Open a new instance of Firefox and access the AFWeb site: • Browse to **http://afweb.training.lab/**. • Click **Cookie Consistency Demo** link. Set the cookie with the custom value: • Enter **<your name>** in the Name field. • Click **Submit**. This value was accepted successfully and your custom value is displayed in the page.
4.	Use Web Developer Extension to view the current cookies on the site: • Click **Tools > Web Developer Extension > Cookies > View Cookie Information**. • Switch to the Cookie Information tab and confirm only two (2) cookies exist: citrix_ns_id This is the NetScaler sessionization tracking cookie. MySiteVisitorName Persistent cookie created by the AFWeb application. View the Cookie details for MySiteVisitorName: • Notice that this time the Cookie value is encrypted and you cannot identify your original value. • Notice that the cookie contains an HTTPOnly flag. • Attempts to edit the cookie using Web Developer Extension will be prevented by the browser. The Session Cookie ASPSESSIONID is not displayed at all client side as it is being proxied by the NetScaler. Also, the NetScaler does not generate tracking cookies when using the Transform option for Cookie Consistency checks.
5.	Return to the AFWeb tab. • Click **Modify** to modify the cookie. The cookie is not changed by client side code within the browser.
6.	Try to hack the cookie using Tamper Data: • In Firefox, click **Tools > Tamper Data**. • In Tamper Data, click **Start Tamper**. Return to AFWeb tab in Firefox: • Click **Modify**. Switch to Tamper Data: • Click **Tamper**. • Find the cookie named **MySiteVisitorName** in the Cookie field on the Request Header Name column (left pane). All cookies are listed in this one field; you will need to use the cursor to navigate the field contents. Manually change the MySiteVisitorName string to: **`MySiteVisitorName="baduser"`** • Click **OK** to submit the value. NetScaler identified this request as a violation and removed the cookie from the request. Verify you are back at the prompt for your name, indicating no cookie was sent to server.

7.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm a Cookie Consistency check violation was identified for the MySiteVisitorName cookie. Syslog indicates the action taken was <blocked>. Also note Cookie Transformations are not logged in syslog.
8.	Clear all cookies to fully reset demonstration before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate: Click **History > Clear Recent History**. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox (including Tamper Data), then re-open Firefox, before continuing. This ensures any session cookies are also cleared.

Cookie Consistency 4 - Configure Relaxation

Step	Action
1.	Return to the NetScaler Configuration utility and access the AFWeb Profile. • Navigate to **Security > Application Firewall > Profiles**. • Select **appfw_prof_afweb** and click **Edit**.
2.	Create an exemption allowing the MySiteVisitorName cookie to be modified client-side. Update the AFWeb Profile: • Click **Relaxation Rules** under Advanced Settings. • Select (check) **Cookie Consistency** under Relaxation Rules and click **Edit**. • Click **Add**. • Enter **MySiteVisitorName** in the Cookie Name field. • Click **Create**. Click **Close** to apply the Cookie Consistency Relaxation Rules settings. Click **Done** to close the profile.

3.	Switch to Firefox and access the AFWeb site: • Browse to **http://afweb.training.lab/**. • Click **Cookie Consistency Demo**. Set cookie with your custom value: • Enter **<your name>** in the name field. • Click **Submit**. Verify your custom value is displayed. • Click **Modify**. Verify the cookie was updated and "Modified by Client" is now displayed. • Click **Delete**. Verify the cookie was successfully deleted by the application. No violation was detected once the cookie was exempted from the protection.
4.	Save the NetScaler Configuration.

Takeaways:

- Cookie Consistency Check requires the NetScaler to utilize sessionization and session tracking for each user connecting to the application. With the block protection enabled, the NetScaler will track the sessionization cookie assigned to the user's session and all persistent and all session cookies generated by the application.
- In order for Cookie Consistency check to be effective, the NetScaler must first see the creation of the original application cookies. Therefore, enabling this security check for users who have existing long-lived, persistent cookies on their devices to begin with due to previous connections to the application will generate numerous false positives for cookie violations. Users must delete existing cookies and request fresh cookies when this security check is enabled. Existing cookies are not tracked by the NetScaler and will be seen as an automatic violation.
- Cookie Consistency check also supports a transform action, which allows three potential modifications to the cookie set by the application. When the transform action is enabled, the block behavior and conditions for violation are slightly different than when the action is set to block.
  - Encrypt Server Cookies: Encrypt all, Encrypt session only, Decrypt only, none.
  - Proxy Sever Cookies: Session only, none.
  - Flags to add to cookies: All, HTTPOnly, Secure, none.
- With the block action enabled, upon violation, the Application Firewall strips the cookies in violation from the request and the cookie never reaches the server. A block violation will be logged, but no "block" error page will be displayed.
- With the transform action enabled, cookies will be rewritten according to the transformation rules. Cookie transformations are not logged in syslog.
- Block and Transform actions can be enabled at the same time.

Exercise 4-8: Form Field Consistency Check

In this exercise, you will demonstrate how the form field consistency security check can protect against parameter and form field manipulation attacks.

Scenario:

The security team has asked you to implement Form Field Consistency protection for the WebGoat application. But after their initial testing, they were getting some immediate violations. In this case, they want Form Field Consistency to be enabled to protect the website from client-side manipulation

Requirements for this Scenario:

- First, view the attack against WebGoat when the protection is disabled. Use the Parameter Tampering > Exploit Hidden Fields demonstration page for testing.
- Next, explore the normal operation of the website and investigate how the application handles data passing during the purchase process.
- Then, enable the Form Field consistency check. Prior to performing the attack, first attempt to operate the site normally.
    - Identify any violations that are being identified erroneously and update the Form Field Consistency check to allow this content.
    - Verify normal site operations are restored.
- Finally, enable the Form Field Consistency check and ensure the previous attack is prevented.

In this exercise, you will perform the following tasks:

- Form Field Consistency 1 - Parameter and Form Field Manipulation - Attack Demonstration
- Form Field Consistency 2 - Attack Protection

Form Field Consistency 1 - Parameter and Form Field Manipulation - Attack Demonstration

Step	Action
1.	Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI: • Run the following command to disable Application Firewall: `disable ns feature appfw`
2.	In Firefox, browse to http://webgoat.training.lab/WebGoat/attack. • Log on as **guest / guest**. • Click **"Start WebGoat"** on the main page.
3.	Access Exploit Hidden Fields lesson in WebGoat: • Navigate to **Parameter Tampering > Exploit Hidden Fields**.

4.	In Firefox, open Tamper Data • Click **Tools > Tamper Data**. • Click **Start Tamper**. Tamper Data will run in a separate window. Arrange the windows so you can see Tamper Data on one side and your regular Firefox browser running WebGoat on the other. While Tamper Data is running it will intercept requests client-side before submitting them to the server, allowing you to manipulate headers and query parameters in the request.
5.	Switch to WebGoat to perform the exploit: • Enter **10** in the Quantity field. • Click **Purchase**. Use Tamper Data to modify the request: • Click **Tamper** in the Tamper with Request dialog. In the right-pane, Tamper data displays the query parameters posted in the request. Change the following parameters: • Enter **10** in QTY (quantity). • Enter **1.99** in Price. • Click **OK** to submit the modified request. • Confirm the total price displayed reflects the modified values and you received 10 TV's for $19.90. Return to Tamper Data and stop tamper: • Click **Stop Tamper**. • Keep Tamper Data open.
6.	Switch to WebGoat in Firefox and click **Restart this Lesson** before proceeding.

Form Field Consistency 2 - Attack Protection

Step	Action
1.	Switch to **Putty (2)** and re-enable the Application Firewall feature using the NetScaler CLI: • Run the following command to enable Application Firewall: `enable ns feature appfw` **NOTE**: Form Field Consistency check is currently disabled and not yet in effect.

2.	Switch to Firefox and browse to WebGoat: http://webgoat.training.lab/WebGoat/attack. • Navigate to **Parameter Tampering > Exploit Hidden Fields**. • Refresh or reload the page. Use Web Developer Extension to view Form Fields: • In Firefox, Click **Tools > Web Developer Extension > Forms > Display Form Details**. • Notice that the hidden field "Price" is displayed using the Web Developer Extension Plugin. **NOTE**: • Hidden Fields can be viewed in the page source, as well. • Web Developer Extension (and similar tools) could be used to manipulate this form field; however, WebGoat has some built in protections against modifying the price in this way, so the hack demonstrated using Tamper Data to adjust the Query Parameters during the POST is the optimal attack for this site.
3.	Return to the NetScaler configuration utility and connect to the NSIP: http://192.168.10.101.
4.	Update the Application Firewall profile for WebGoat: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_webgoat** and click **Edit**. Enable the Form Field Consistency security check: • Click **Security Checks** under Advanced Settings. • Select (check) **Form Field Consistency** under Security Checks and click **Action Settings**. • Enable (check) **Block**. • Enable (check) **Log**. • Enable (check) **Stats**. • Click **OK** to apply the Form Field Consistency settings. Click **OK** at the bottom of Security Checks to apply changes to the profile.
5.	Switch to Firefox and browse to WebGoat: http://webgoat.training.lab/WebGoat/attack. • Navigate to **Parameter Tampering > Exploit Hidden Fields**. • Reload the page if it was already displayed. Expected Result: WebGoat is blocked by the Application Firewall, but no attack as been performed, yet. (In a few odd situations, WebGoat may not be blocked immediately. Double-check if Form Field Consistency check is enabled within the profile. If still an issue, reset Firefox before continuing.) Explanation: Parts of the WebGoat navigation menu are modified using client-side code. This can result in the regular navigation parameters like "menu" and "screen" triggering a false positive once Form Field Consistency is enabled. Before proceeding with the exercise, you will configure an exception for these particular parameters in order to allow legitimate site operation.

6.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: The syslog output identifies the violation for Form Field Consistency check violation for field menu. **Note**: Syslog only flagged the first violation as the request was redirected without requiring additional processing. The "Screen" parameter will also trigger a violation. Both updates will be made at one time.
7.	Return to the NetScaler configuration utility and connect to the NSIP: http://192.168.10.101.
8.	Update the Application Firewall profile for WebGoat with a relaxation for the Form Field Consistency Check. • Remain in the profile settings for appfw_prof_webgoat. • Click **Relaxation Rules** under Advanced Settings. • Select (check) **Form Field Consistency** under Relaxation Rules and click **Edit**. Create exemption for the Menu form field (parameter): • Click **Add**. • Enter **menu** in the Form Field Name. • Enter **^http://webgoat[.]training[.]lab/WebGoat/attack** in the Action URL. (Notice: No end-of-line anchor is specified.) • Click **Create**. Create exemption for the Screen form field (parameter): • Click **Add**. • Enter **screen** in the Form Field Name. • Enter **^http://webgoat[.]training[.]lab/WebGoat/attack** in the Action URL. • Click **Create**. Click **Close** to close the Form Field Consistency Relaxation Rules window. **NOTE**: For the Action URL regular expressions, **DO NOT** include a trailing slash or "$". This URL pattern will exempt the Screen and Menu parameters from every URL on webgoat.training.lab. This is necessary due to how WebGoat navigation is handled.
9.	Switch to Firefox and browse to WebGoat: http://webgoat.training.lab/WebGoat/attack. • Navigate to **Parameter Tampering > Exploit Hidden Fields**. Use Web Developer Extension to view Form Fields: • In Firefox, Click **Tools > Web Developer Extension > Forms > Display Form Details**. • Notice that the hidden field "Price" is displayed using the Web Developer Plugin. • Notice that an additional hidden field "as_fid" has been added to the page once the Form Field Consistency check was enabled. Refresh the page when done viewing the Form Field properties.

10.	Perform the Hidden Field Manipulation / Form Field Manipulation attack: Switch to the Tamper Data window: • Click **Start Tamper**.
11.	Switch to WebGoat to perform the exploit: • Enter **10** in the quantity field. • Click **Purchase**. Use Tamper Data to modify the request: • Click **Tamper** in the Tamper with Request dialog. In the right-pane, Tamper data displays the query parameters posted in the request. Change the following parameters: • Enter **10** in QTY (quantity). • Enter **1.99** in Price. • Click **OK** to submit the modified request. Confirm the attack was blocked by Application Firewall. Return to Tamper Data and stop tamper: • Click **Stop Tamper**. • Close the Tamper Data window.
12.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm syslog identified a Field Consistency violation and identified the violation affecting the price field.
13.	Close the Application Firewall profile for WebGoat. Click **Done**.
14.	Save the NetScaler configuration.

Takeaways:

- Form Field Consistency check is not enabled by default with the basic protections, but is enabled with the advanced protections.
- The NetScaler takes forms in responses with a custom value in the as_fid field and then compares the computed value of the form fields with the form is represented by the client during a request. If the current form computed value doesn't equate to the original computed value or if the as_fid field is present, a violation occurs.
- Form Field consistency check ensures that:
  - No new fields are created or no fields were removed from the original response.
  - Hidden and read-only fields do not change values.
  - Fields should not change types.
  - Field lengths should not change, if originally specified.
  - Field values are consistent with the field input type.

Exercise 4-9: Credit Card and Safe Objects

In this exercise, you will demonstrate response time protections and prevent credit card (PCI) and other sensitive information leaks from being displayed in web responses.

Scenario:

The security wants to ensure that the Application Firewall can also prevent leakage of protected information for their applications that accept payment information and sensitive user identification information. Specifically, they want to confirm the ability to block credit card numbers, US tax identification (social security) numbers, and phone numbers in a variety of formats.

Requirements for this scenario:

- Use AFWeb to display Credit Card, US Phone, and US Tax ID numbers.
- Configure the NetScaler to block on violation.
- Then update the protection to transform on violation and demonstrate the alternate attack mitigation responses for X-out and Remove.

In this exercise, you will perform the following tasks:

- Credit Card Exploit and Protection
- SafeObject Exploit and Protection

Credit Card Exploit and Protection

Step	Action
1.	Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI: • Run the following command to disable Application Firewall: `disable ns feature appfw`
2.	In Firefox, browse to http://afweb.training.lab. • Click **Credit Card Demo** link. • Verify a list of Credit Card numbers are displayed. **NOTE**: While the SQL Injection attack in WebGoat exposes a list of credit card numbers in the user property dump, none of the numbers in the WebGoat application actually conform to valid test credit card numbers and will not be suitable for the attack prevention demonstration with Application Firewall. AFWeb will be used for this demonstration. Please note that the numbers listed in AFWeb consist of a mixture of invalid patterns and numbers that conform to test accounts.
3.	Switch to **Putty (2)** and re-enable the Application Firewall feature using the NetScaler CLI: • Run the following command to enable Application Firewall: `enable ns feature appfw`
4.	Return to the NetScaler Configuration Utility: http://192.168.10.101/.

5.	Update the AFWeb profile to enable Credit Card protection: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_afweb** and click **Edit**. Update the Security Check settings: • Click **Security Checks** under Advanced Settings. • Select (check) **Credit Card** and click **Action Settings**. • Enable (check) **Block**, **Log**, **Stats**. • Verify Maximum credit cards allowed per page is set to 0 (no credit cards). • Enable (check) all credit card types under Protected Credit Cards. Click **OK** to apply the security check settings.
6.	View the AFWeb Profile Settings: • Click **Profile Settings** under Advanced Settings. • Scroll down to Common Settings. • Verify **Secure Credit Card Logging** is enabled. Click **OK** to close the Profile Settings page. **NOTE**: Secure Credit Card Logging is enabled by default. This allows Credit Card protection violations to be logged, but prevents logging the credit card numbers to the syslog output. It is strongly recommended that this value is NOT disabled.
7.	Switch to Firefox, and return to AFWeb: • Browse to **http://afweb.training.lab/**. • Click **Credit Card Demo**. Verify the connection request was terminated and the page does not load. Response-time security checks terminate the response; they do not redirect to the HTML error page.
8.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm a credit card violation is observed (APPFW_SAFECOMMERCE) and that the action taken was blocked. Notice that the credit card numbers are not logged to syslog. With Secure Logging of Credit Card numbers enabled (GUI or CLI setting), credit card numbers are NOT included; if security logging of Credit Card numbers is disabled, Credit Card numbers will be included in syslog events. (Secure Logging is on by default; this is a profile setting.)
9.	Return to the NetScaler Configuration Utility: http://192.168.10.101/.

10.	Update the AFWeb profile to enable Credit Card protection using transform actions: - Navigate to **Security > Application Firewall > Profiles**. - Select **appfw_prof_afweb** and click **Edit**. Update the Security Check settings: - Click **Security Checks** under Advanced Settings. - Select (check) **Credit Card** and click **Action Settings**. - Disable (uncheck) **Block**. - Enable (check) **X-out**. Click **OK** to apply the security check settings. Click **Done** to close the profile.
11.	Switch to Firefox, and return to AFweb: http://afweb.training.lab/. - Click **Credit Card Demo**. This time the response is displayed but numbers that match valid Credit Card patterns are X-ed out except for the last 4 digits. Numbers not conforming to valid Credit Card patterns are not masked. Notice that certain credit card patterns can be identified even if digits are separated by spaces ( ) or hyphens (-).
12.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm a credit card violation is observed (APPFW_SAFECOMMERCE) and that the action taken was transformed. Notice that the credit card numbers are not logged to syslog.

SafeObject Exploit and Protection

Step	Action
1.	Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI: - Run the following command to disable Application Firewall: `disable ns feature appfw`
2.	In Firefox, browse to http://afweb.training.lab. - Click **Safe Object Demo** link. - Verify a list of US Social Security Numbers and Phone Numbers are displayed.
3.	Switch to **Putty (2)** and re-enable the Application Firewall feature using the NetScaler CLI: - Run the following command to enable Application Firewall: `enable ns feature appfw`
4.	Return to the NetScaler Configuration Utility: http://192.168.10.101/.

5.	Update the AFWeb profile to enable Safe Object protections: - Navigate to **Security > Application Firewall > Profiles**. - Select **appfw_prof_afweb** and click **Edit**. Update the Security Check Relaxation Rule settings: - Click **Relaxation Rules** under Advanced Settings. - Select (check) **Safe Object** and click **Edit**.
6.	Create a Safe Object for US Tax ID number patterns: - Click **Add**. - Enter **US TaxID** in the Safe Object Name field. - Enable (check) actions **Block**, **Log**, **Stats**. - Enter **11** for Maximum Match Length. - Enter the following regular expression: `\d{3}-\d{2}-\d{4}` Click **Create** to configure the Safe Object. **NOTE**: Other regular expressions could be used depending on the complexity of the pattern being matched. Watch out for expressions being too general and matching a wide range of legitimate page content. Also note that each Safe Object is, in effect, its own security check. Each object pattern, can be protected with its own unique settings for Block, Log, Stats or transform options for X-Out or Remove.
7.	Create a Safe Object for US Phone Number patterns: - Click **Add**. - Enter **US Phone** in the Safe Object Name field. - Do not enable **Block**. - Enable (check) actions **Log**, **Stats, X-Out**. - Enter **12** for Maximum Match Length. - Enter the following regular expression: `\d{3}-\d{3}-\d{4}` Click **Create** to configure the Safe Object. Click **Close** to close the Safe Object Rules window. **NOTE**: Other regular expressions could be used depending on the complexity of the pattern being matched.
8.	Switch to Firefox, and return to AFWeb: http://afweb.training.lab/. - Click **Safe Object Demo**. Verify the connection request was terminated and the page does not load. Response-time security checks terminate the response; they do not redirect to the HTML error page. Since the page contained both objects and one is set to block, while the other is set to transform, the block action takes precedence and no content is displayed.

9.	Switch to **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm a Safe Object violation is observed (APPFW_SAFEOBJECT) and that the action taken was blocked. The security check does identify the security check (Safe Object), the safe object name the pattern matched, and the pattern found. **NOTE**: It is important to note that unlike with Credit Cards, safe object patterns are included in the log and there is no "secure logging" enabled/disabled equivalent option. The security check logging action can be disabled to prevent logging content, but this omits a record of the action as well.
10.	Return to the NetScaler Configuration Utility: http://192.168.10.101/.
11.	Update the SAFE Object Security Check settings: • Select (check) **Safe Object** under Relaxation Rules and click **Edit**. Update the US TaxID safe object: • Select (check) **US TaxID** and click **Edit**. • Disable (uncheck) **Block**. • Enable (check) **Remove**. • Click **OK** to close the Safe Object rule. Click **Close** to close the Safe Object rules list. Click **Done** to close the profile properties.
12.	Save the NetScaler configuration.
13.	Switch to Firefox, and return to AFweb: http://afweb.training.lab/. • Click **Safe Object Demo**. This time the response is displayed. • US Tax ID numbers have been removed from the response. • US Phone Numbers have been x-ed out (notice this is a full pattern mask and not a partial mask.)
14.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: The SafeObject event is logged (for at least one item) and the action indicated is transformed. **NOTE**: To avoid logging safe objects in syslog, you will need to disable the log action for the Safe Object. This means you won't have the record of the security check action when violations occur.

Takeaways:

- Response time checks will terminate the response upon violation when the block action is specified.
- For Credit Cards, the response time check can allow administrators to specify the credit card patterns to protect (within the list of providers supported by NetScaler), the maximum allowed number of cards permitted per page, and whether to block on violation or X-out on violation.
- By default Credit Card numbers are not logged to syslog when an event is reported. This is controlled by the "Secure Credit Card Logging" parameter in each application firewall profile. Secure logging is enabled by default. If the setting is disabled, credit card numbers which trigger violations will be included in syslog events.
- Safe Object is a custom response time security check. Administrators can define a regular expression that identifies content of concern. Each Safe Object rule acts as its own security check and can be individually configured for block, log, stat, x-out, and remove behavior.
    - X-out transformation performs a full mask of the pattern and not a partial mask. The exact behavior depends on how the regex is defined.
    - If multiple objects appear in the same page with different violations in effect, Block will take precedence over transformations and the response will be terminated.
    - If multiple objects appear in the same page and they all use transformation actions such as X-out or remove, then each object will be handled according to its protection settings.
    - If LOG is enabled, the violation will appear in syslog but so does the Safe Object content that triggered the violation. For sensitive information, the log action may need to be disabled to avoid including this information in syslog as another point of exposure.

Exercise 4-10: Start URLs with URL Closure / Learning

In this exercise, you will configure application firewall protections for AFWeb to use Start URLs with URL Closure.

Scenario:

The application team for AFWeb has brought up some concerns with the security team. They realize that a portion of their site is not properly implementing access security. Content available at the /private2.htm page should only be available to users who properly navigate various gatekeeping pages first, but they want the content denied if someone attempts to bypass this. Placing the content on a Deny URL list for everyone would be too strict to allow legitimate site operation.

The application team wants the Application Firewall configuration to be updated to allow for this content. They also want to verify that future site updates can be identified and accounted for by quickly updating the site.

The security team wants you to implement URL Closure and demonstrate the Application Firewall learning engine to assist with the re-configuration of the site, using these new settings. This way they can see how to use learning for future site update cycles.

Requirements for this scenario:

- AFWeb will be used for this scenario.
- Modify the Start URL protections to include URL Closure.
  - Disable the default Start URLs but keep the custom URLs configured during the initial configuration.
- Configure the Start URLs so that regular site navigation succeeds without generating errors, but still prevent direct access to site URLs not covered by normal site navigation.
  - Access to /private.htm should be blocked at all times, even if not included on the Deny URL list.
  - Access to /private2.htm should be denied if directly accessed, but allowed if the website directs users to this link.

In this exercise, you will perform the following tasks:

- Configure Start URLs with URL Closure and Learning (AFWeb)

Configure Start URLs with URL Closure and Learning (AFWeb)

Step	Action
1.	Switch to **Putty (2)** and enable the Application Firewall feature using the NetScaler CLI: • Run the following command to enable Application Firewall: `enable ns feature appfw` At this point, the AppFw Profile for AFWeb is configured with Start URLs enabled and URL Closure Off. The current behavior of AFWeb is determined by the current list of Start URLs and Deny URLs defined.

2.	In Firefox, browse to http://afweb.training.lab Test the following demonstration links on the default page and confirm the results: • **Allow Demo**: Link takes you to the /allow.demo page. Link is allowed due to previous Start URL configuration. • **Deny Demo**: Link attempts to take you to the /denyme.htm page, but is blocked due to the previous Deny URL configuration. • **URL Closure Demo**: Link takes you to /closure1.htm. **Closure2 Demo**: Link takes you to /closure2.htm. **Private2 Demo**: Link takes you to /private2.htm. All three links are allowed as the URLs conform to the normal Start URL patterns currently configured. Manually browse to the following URL paths and confirm the results: • Manually browse to **http://afweb.training.lab/private.htm**. This is denied due to the previous Deny URL configuration. • Manually browse to **http://afweb.training.lab/private2.htm**: This is allowed. This step demonstrates which URLs are allowed or denied based on the current Start URL (without URL Closure configuration).
3.	Return to the NetScaler Configuration Utility in Chrome. Connect to NS_VPX_01 using the NSIP at http://192.168.10.101.
4.	Update the Application Firewall profile for AFWeb: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_afweb** and click **Edit**. Update the Security Check to enable Learnings: • Click **Security Checks** under Advanced Settings. • Enable (check) **Learn** on the following security checks only: **Start URL** **Cookie Consistency** **Field Formats** **HTML Cross-Site Scripting** **HTML SQL Injection** • Verify Learning is disabled on: Credit Card Content-Type CSRF Form Tagging Click **OK** to apply settings. Click **OK**, if prompted, to confirm "Form Tagging has been enabled." Keep the AFWeb profile open.
5.	Update the AFWeb Profile to enable URL Closure: • Select (check) **Start URL** under Security Checks and click **Action Settings**. • Enable (check) **Enforce URL Closure**. Click **OK** to close the Start URL Settings.

6. Update the AFWeb Profile relaxation rules for Start URLs:

- Click **Relaxation Rules** under Advanced Settings.
- Select (check) **Start URL** under Relaxation Rules and click **Edit**.

Disable the default Start URLs initially configured by the basic profile.

- Select (check) the following rule and click **Disable** and click **Yes** to confirm.

```
^[^?]+[.](html?|shtml|js|gif|jpg|jpeg|png|swf|pif|pdf|css|csv)
$
```

- Select (check) the following rule and click **Disable** and click **Yes** to confirm.

```
^[^?]+[.](cgi|aspx?|jsp|php|pl)([?].*)?$
```

The following rules (4) will remain enabled. Notice that these rules only allow access to the basic AFWeb content:

- Basic AFWeb FQDN

```
^http://afweb[.]training[.]lab/$
```

- Basic AFWeb VIP

```
^http://172\.21\.10\.111/$
```

- Content with the .ico extension

```
^[^?]+[.]ico$
```

- The allow.demo page

```
/allow[.]demo$
```

Click **Close** to close the Start URL Relaxation Rules summary.

NOTE: The two default Start URLs broadly allow a wide-range of content and therefore should never be used when URL Closure is enabled. As a best practice, the default URLs should be removed from an existing profile when enabling URL Closure or start with a new Advanced profile instead. However, for lab purposes the rules will be disabled, instead of deleted, to allow students the opportunity to review or restore previous settings after this exercise.

7. Update the AFWeb Profile rules for Deny URLs:

- Deselect **Start URL** under Relaxation Rules.
- Select (check) **Deny URL** under Relaxation Rules and click **Edit**.
- Change the Items per page from 25 to 250, if needed.
- Select (check) the rule to block **/private[.]htm** and click **Disable** and **Yes** to Confirm.

Click **Close** to close the Deny URL Relaxation rules summary.

Keep the AFWeb profile properties open.

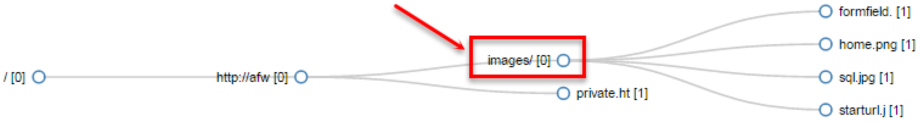
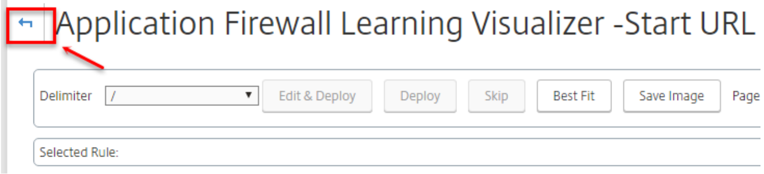
8. Clear all session state in Firefox to fully reset demonstration:

- Switch to Firefox.
- Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate method: click **History > Clear Recent History**.
- Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details.
- Click **Clear Now**.

IMPORTANT:

- Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared.

9.	Test how the current Start URLs with URL Closure works with the AFWeb page. Use the following procedure to test AFWeb and generate learned data for additional Start URL requirements. • In Firefox browse to **http://afweb.training.lab/**. Notice not all graphics are returned.
10.	Switch to **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Violations for various graphics are displayed due to URL Closure and changes to the Start URLs.
11.	Return to Firefox. Click on the following links to navigate to the site, return to http://afweb.training.lab after each test. It is important that you browse the site from the pages listed and do not manually navigate the site by manually entering the URLs: • **Allow Demo**. • **Deny Demo**. • **SQL Injection Demo**. Enter **test** in the lookup value field and click **Submit** to navigate to /sql.asp. • **Form Field Demo**. Select **Standard Checking** in account type drop-down list and click **Submit**. Select **Standard Checking** in account type again and click **Submit** on the /field.asp page. • **URL Closure Demo > Closure2 Demo > Private2 Demo**. Manually browse to the following URLs: • Browse to **http://afweb.training.lab/blocked.htm**. • Browse to **http://afweb.training.lab/private.htm**. • Browse to **http://afweb.training.lab/private2.htm**. The current Start URL list does not completely allow access to all required elements of AFWeb, after the initial test case. Learning will be used to update the required rules. Learning will identify violations not covered by the current URL Closure rules.
12.	Return to the NetScaler configuration utility in Chrome and connect to the NSIP: http://192.168.10.101.
13.	View the learned rules in the Application Firewall profile for AFWeb: • Click **Learned Rules** under Advanced Settings. • Select (check) **Start URL** under Learned Rules and click **Edit**. View the additional rules identified by Learning with URL Closure enabled: • Four of the rules are dependencies on jpg or png in the /images directory. (You may have more objects, if additional pages were tested.) • One of the rules identifies the attempt to direct browse to /private.htm. **Do not** deploy any rules at this time. Click **Close** on the Start URL Learn Rules summary.

14.	Use the Learning Visualizer to view the learned rules and deploy settings: • Select (check) **Start URL** under Learned Rules. • Click **Visualizer**. View the rules in the visualizer: • The three graphics are displayed under the images/ node. • The /private.htm object is displayed from the root http//afweb.training.lab node. Use the visualizer to generate a consolidated regular expression and deploy the custom rule to the Start URL list: • Click on the **images/** node in the visualizer graphic:  / [0] http://afw [0] images/ [0] private.ht [1] formfield. [1] home.png [1] sql.jpg [1] starturl.j [1] • Verify a consolidated rule similar to the following is displayed in the Selected Rule field. This rule should allow content from the images/ directory only. (There may be slight variations to the expression if you tested more links.) Final output should be similar to: **`http:\/\/afweb\.training\.lab\/images\/[\.ad-jl-u]{7,13}`** • Click **Deploy** to deploy the rule as is to the Start URL list and click **Yes** to confirm. Click the **Back** icon to return to the profile properties:  Application Firewall Learning Visualizer -Start URL Delimiter / Edit & Deploy Deploy Skip Best Fit Save Image Page Selected Rule:
15.	View the updated Relaxation Rules for Start URL: • Click **Relaxation Rules** under Advanced Settings (if the category is not already in the configuration pane). • Deselect (uncheck) **Deny URL**. • Select (check) **Start URL** under Relaxation Rules and click **Edit**. • Verify the rule to allow /images content from the learning engine was deployed to the Start URL list and is enabled. **`http:\/\/afweb\.training\.lab\/images\/[\.ad-jl-u]{7,13}`** Click **Close** to close the Start URL Relaxation Rules summary.
16.	Click **Done** to close the AFWeb profile.
17.	Save the NetScaler configuration.

18.	Clear all session state in Firefox before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate method: click **History > Clear Recent History**. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared. • Re-open Firefox. **Do not** browse to http://afweb.training.lab yet.
19.	Demonstrate how URL Closure plus the initial Start URLs successfully allow access to the linked parts of the site: • Switch to Firefox, browse to **http://afweb.training.lab**. URL succeeds. Test the following demonstration links from the default page and confirm the results. Return to the http://afweb.trainign.lab between tests. • **Allow Demo**: Link takes you to the /allow.demo page. Link is allowed due to previous Start URL configuration. • **Deny Demo**: Link is still blocked due to the previous Deny URL configuration. • **URL Closure Demo**: Link takes you to /closure1.htm. **Closure2 Demo**: Link takes you to /closure2.htm. **Private2 Demo**: Link takes you to /private2.htm. All three links are allowed as the URLs conform to the normal Start URL patterns currently configured. Manually browse to the following URL paths and confirm the results: • Manually browse to **http://afweb.training.lab/private.htm**. This is denied due to URL Closure. • Manually browse to **http://afweb.training.lab/private2.htm**: This is allowed since you had previously been allowed to navigate via the URL Closure Demo links.
20.	Clear all session state in Firefox before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate method: click **History > Clear Recent History**. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared. • Re-open Firefox. **Do not** browse to http://afweb.training.lab yet.

21.	Attempt to direct browse to the /private2.htm without going to the default page first: For each of the following tests, type in the URL manually. **Do not** connect to the root page (http://afweb.training.lab/) first. Test the following URLs: • Manually browse to **http://afweb.training.lab/private2.htm**. This attempt is blocked. • Manually browse to **http://afweb.training.lab/sql.asp**. This attempt is blocked. • Manually browse to **http://afweb.training.lab/field.asp**. This attempt is blocked. All three URLs are blocked as you did not start at a designated Start URL first. **NOTE**: If you are watching the violations in syslog (Putty (1)), these will appear as Start URL violations in syslog.
22.	Confirm URLs are allowed if covered by URL Closure: • Manually browse to **http://afweb.training.lab/**. Use the navigation links to access the specified URLs. Return to the (/) between each test case: • **SQL Injection Demo**. This loads the /sql.htm page. Enter **test** in the Lookup Value field and click **Submit**. The /sql.asp page displays successfully. • **Form Field Demo**. This load the /field.htm page. Click **Submit**. The /field.asp page displays successfully. • **URL Closure Demo** > **Closure2 Demo > Private2 Demo**. This /private2.htm page displays successfully and is not blocked if access from an allowed Start URL and links covered by URL Closure. Attempts to direct browse to /private2.htm will fail, if not covered by closure. **NOTE**: If you are watching the violations in syslog (Putty (1)), no additional violations should occur for navigating these links as these pages were covered by URL Closure.
23.	Save the NetScaler configuration.

Takeaways:

- URL Closure changes the function of Start URLs from a whitelist function to a list of allowed entry points for the site. Navigation is only allowed via identified entry points or links supplied to users during requests to an allowed entry point. URL Closure allows users to successfully navigate to all links provided from entry point URLs and sites descended from these allowed entry points.
- The default URLs included in the basic profile should not be used with URL Closure enabled.
- URL Closure can be used to selectively block content based on context. If users navigate to authorized Start URLs, URL Closure will allow navigation to dependent links. However, attempts to directly browse to sites not covered by URL Closure will be denied.
- The NetScaler learning engine learns violations and is therefore dependent on the security state's configuration for what will be identified.
    - It is recommended that minimal initial configurations are made prior to enabling Learning.
    - For example, enable URL Closure and at a minimum the root web site URL to provide minimum required access before performing Learning.
- During learning phases, security checks can be set to disable blocking so the Application Firewall functions in observation mode.

Exercise 4-11: CSRF Form Tagging / Referer Header Validation

In this exercise, you will use Start URLs with URL Closure and Referer header validation to prevent a type of CSRF attack.

Scenario:

The WebGoat application has a newly discovered vulnerability that could allow a CSRF attack from a remote site, if URLs are called with manipulated query parameters.

To protect against this attack from a remote website, URL Closure with Referer header validation will be enabled for WebGoat.

However, due to the way that WebGoat constructs legitimate site URLs, the Application Profile for WebGoat will have to have Form Field Consistency checks disabled, in order to demonstrate the specific attack and protection in mind. Under normal operations, Referer header validation and Form Field Consistency checks should remain enabled to provide a range of protections.

During the attack, the goal is to get a remote site (AFWeb) to issue a custom URL to WebGoat that will result in the transfer of secure funds. This will require that URL Closure is properly configured for WebGoat, Form Tagging is enabled, and Referer header validation is enabled.

Requirements for this scenario:

- First, this exercise will demonstrate the attack from a rogue server (AFWeb) against WebGoat while no application firewall protections are enabled.
- Next, the WebGoat Application Firewall profile will be updated so that URL Closure is enabled. Learning will also be enabled to quickly update the required Start URLs with additional patterns. At the end of this task, verify the following:
  - Ensure that the Start URLs with URL Closure are properly configured to allow successful navigation of WebGoat's regular pages.
  - Disable Form Field Consistency, to prevent conflicts with regular site navigation and the CSRF attack demonstration.
- Then, repeat the CSRF attack with URL Closure enabled (but no Referer Header Validation protection). This will show the difference in URL Closure protection vs. Referer Header Validation.
- Finally, demonstrate the attack prevention using Referer Header validation on WebGoat.

In this exercise, you will perform the following tasks:

- CSRF 1: Demonstrate CSRF/Referer Header Attack
- CSRF 2: Configure URL Closure for WebGoat
- CSRF 3: Test URL Closure without Referer Header Validation Protection
- CSRF 4: Test Referer Header Validation Protection to Prevent the CSRF Attack

CSRF 1: Demonstrate CSRF/Referer Header Attack

Step	Action
1.	Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI: • Run the following command to disable Application Firewall: `disable ns feature appfw`

2.

Open Firefox and browse to **http://webgoat.training.lab/WebGoat/attack**.

- Log on as guest / guest, if prompted.
- Click **Start WebGoat**, if required.

3.

In WebGoat (Tab 1), view the CSRF attack lesson:

- Navigate to **Cross-Site Scripting (XSS) > Cross Site Request Forgery (CSRF)**.

The objective of this attack is to get a URL to submit data to this page and include the "transferfunds=4000" as an additional query parameter. For our lab exercise, we are going to initiate the attack against WebGoat from AFWeb.

Take note of the WebGoat URL that corresponds to the CSRF attack page. Due to the way WebGoat works, the Screen query parameter will vary per student. Take note of the Screen value in the URL for your instance of WebGoat. Copy the URL to notepad or write it down.

```
http://webgoat.training.lab/WebGoat/attack?Screen=XXXX
&menu=900
```

Screen ID:________________________________

NOTE: The WebGoat load balancing virtual server is using a 24-hour long persistence value. However, if you repeat the test multiple times or reset Firefox, the URL query parameter changes. If you are having issues with this exercise, return to this page in WebGoat and confirm the parameter value before performing the attack.

4.

In Firefox, open a new tab (**Tab 2**) and browse to **http://afweb.training.lab**.

- Click **CSRF Demo**.

Enter the following text in the Custom Message field (or copy and paste from within the page):

```
<a href="http://webgoat.training.lab/WebGoat/attack?
Screen=XXXX&menu900&transferFunds=4000">Safe to Click</a>
```

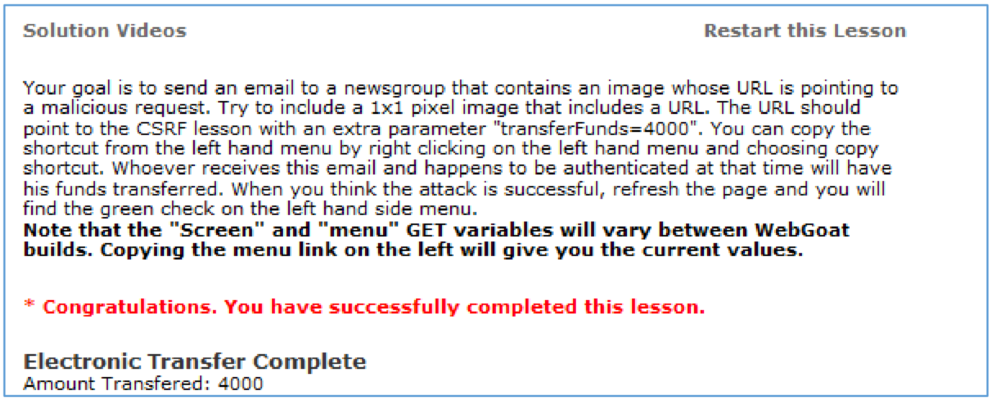
Replace the **XXXX** with the correct set of numbers for your instance of WebGoat.

Click **Submit**.

5.

Click on the link **Safe to Click**.

This will result in the malicious link navigating to WebGoat's CSRF demonstration page. WebGoat then confirms that you have successfully completed the electronic funds transfer:



Your goal is to send an email to a newsgroup that contains an image whose URL is pointing to a malicious request. Try to include a 1x1 pixel image that includes a URL. The URL should point to the CSRF lesson with an extra parameter "transferFunds=4000". You can copy the shortcut from the left hand menu by right clicking on the left hand menu and choosing copy shortcut. Whoever receives this email and happens to be authenticated at that time will have his funds transferred. When you think the attack is successful, refresh the page and you will find the green check on the left hand side menu.
Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

*** Congratulations. You have successfully completed this lesson.**

Electronic Transfer Complete
Amount Transfered: 4000

6.	In WebGoat, Reset the lesson state before continuing: - Navigate to **Cross-Site Scripting (XSS) > Cross Site Request Forgery (CSRF)** and click **Restart this Lesson**. (You must re-navigate to the page for this to work.) - Verify the lesson state reverts and the green checkbox next to the lesson name in the navigation pane is gone.

CSRF 2: Configure URL Closure for WebGoat

Step	Action
1.	Switch to **Putty (2)** and enable the Application Firewall feature using the NetScaler CLI: - Run the following command to enable Application Firewall: `enable ns feature appfw`
2.	Return to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101.
3.	Disable the App Firewall policy on AFWeb (temporarily): - Navigate to **Security > Application Firewall > Policies > Firewall**. - Select (check) **appfw_pol_afweb** and click **Edit**. - Change the expression from true to **false**. Click **OK** to close the policy. **NOTE**: This is being done to temporarily disable AFWeb from protection with Application Firewall and URL Closure, without unbinding policies or changing the last state configured. The Application Firewall protections will be restored after this exercise. For this exercise, AFWeb is acting as a rogue website and is being used to attack WebGoat.
4.	Update the WebGoat profile: - Navigate to **Security > Application Firewall > Profiles**. - Select (check) **appfw_prof_webgoat** and click **Edit**.
5.	Update the Security Check to enable Learnings: - Click **Security Checks** under Advanced Settings. - Enable (check) **Learn** on the following security checks only: **Start URL**
6.	Enable URL Closure: - Select **Start URL** under Security Checks and click **Action Settings**. - Enable (check) **Enforce URL Closure**. Click **OK** to close the Start URL Settings.

<table>
<tr>
<td>7.</td>
<td>Disable Form Field Consistency Protection:
• Disable (uncheck) <b>Block</b>, <b>Log</b>, and <b>Stats</b> next to Form Field Consistency check in the Security Checks summary view.
• Click <b>OK</b> under Security Checks to apply settings.

Keep the WebGoat Profile open.

<b>NOTE:</b> Due to some of the dynamic page IDs and handling of query parameters in WebGoat, the Form Field Consistency check would prevent part of the CSRF attack demonstration and prevent the demonstration of Referer Header validation. The Form Field Consistency check will be disabled for this exercise.

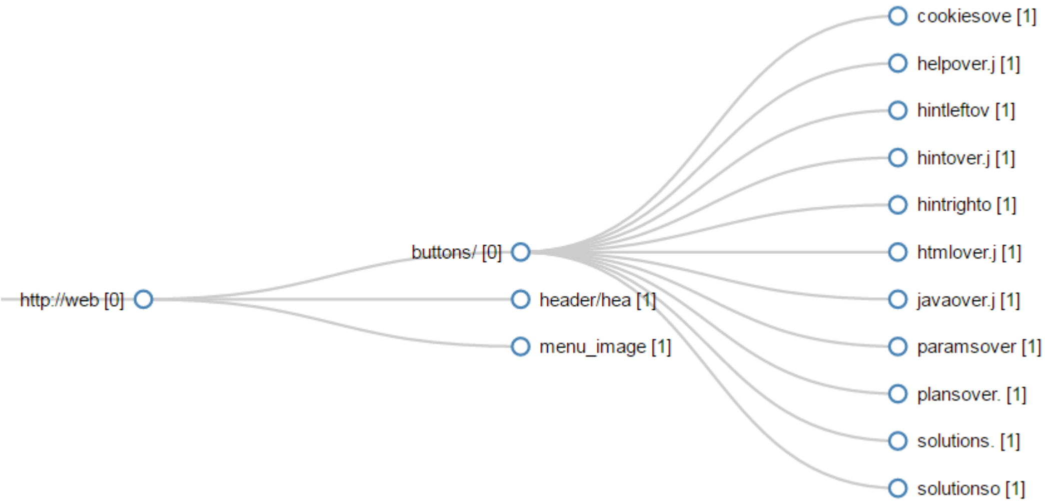
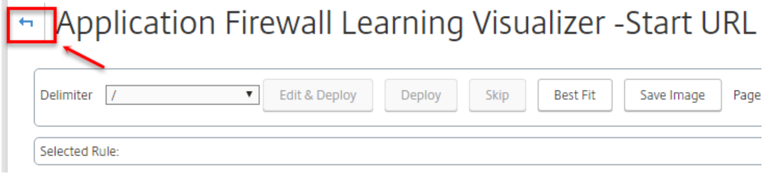
In production, both security checks can and should be used concurrently.</td>
</tr>
<tr>
<td>8.</td>
<td>Update the Start URLs for use with URL Closure:
• Click <b>Relaxation Rules</b> under Advanced Settings.
• Select (check) <b>Start URL</b> under Relaxation Rules and click <b>Edit</b>.

Disable the default Start URLs initially configured by the basic profile.
• Select (check) the following rule and click <b>Disable</b> and <b>Yes</b> to Confirm
<code>^[^?]+[.](html?|shtml|js|gif|jpg|jpeg|png|swf|pif|pdf|css|csv)$</code>
• Select (check) the following rule and click <b>Disable</b> and <b>Yes</b> to Confirm.
<code>^[^?]+[.](cgi|aspx?|jsp|php|pl)([?].*)?$</code>

The following rules (2) will remain enabled. Notice that these rules only allow access to the basic WebGoat content:
• Basic WebGoat FQDN:
<code>^http://webgoat[.]training[.]lab/WebGoat/attack([?].*)?$</code>
• WebGoat favicon:
<code>/favicon[.]ico$</code>

Click <b>Close</b> to close the Start URL Relaxation Rules summary.</td>
</tr>
</table>

9.	Switch to Firefox and test WebGoat. Learning will be used to help identify any additional WebGoat URLs required with the current URL Closure configuration. Access WebGoat using Firefox: • Manually browse to **http://webgoat.training.lab/WebGoat/attack** There will be violations with graphics and other dependent objects at this point. Not all page content will load successfully. Use the WebGoat navigation pane and access the following pages: • Navigate to **Introduction**. • Navigate to **General > HTTP Basics**. • Navigate to **Parameter Tampering > Exploit Hidden Fields**. • Navigate to **Injection Flaws > String SQL Injection**. • Navigate to **Cross-Site Scripting (XSS) > Cross Site Request Forgery (CSRF)**. Take note of the value of the **Screen=XXXX** parameter while here. Screen ID:________________________________ This should generate enough learning data to identify additional Start URL requirements for use with URL Closure.
10.	Return to the NetScaler configuration utility in Chrome: http://192.168.10.101.
11.	View the learned rules within the WebGoat profile: • Click **Learned Rules** under the Advanced Settings. • Select (check) **Start URL** under Learned Rules and click **Edit**. Review the learned rules and confirm that all of the objects are .jpg or .gif images located under the /images/buttons/ directory, the images/headers/ directory or the /images/menu_images/ directory. Therefore, all of this content is safe to allow. Click **Close** to close the Start URL Learn Rules summary.

12.	View the rules in the Visualizer: • Select (check) **Start URL** under Learned Rules and click **Visualizer**. Use the visualizer to consolidate and deploy three rules for the buttons/ directory, the header/ directory, and the menu_image directory: • Click on the **buttons/** node and view the consolidated rule. Click **Deploy** and click **Yes**. • Click on the **header/** node and view the rule. Click **Deploy** and click **Yes**. • Click on the remaining node **http://web[1]** which represents the **menu_image/** node and view the rule. Click **Deploy** and click **Yes**..  http://web [0] buttons/ [0] header/hea [1] menu_image [1] cookiesove [1] helpover.j [1] hintleftov [1] hintover.j [1] hintrighto [1] htmlover.j [1] javaover.j [1] paramsover [1] plansover. [1] solutions. [1] solutionso [1] Click the **Back** icon to return to the profile properties:  Application Firewall Learning Visualizer -Start URL Edit & Deploy Deploy Skip Best Fit Save Image Page Selected Rule: Delimiter / **NOTE**: The exact visualization of the learned rules and expression deployed may vary from the image in this lab guide, depending on the exact test procedure followed.
13.	View the configured Start URL Relaxation Rules: • Select (check) **Start URL** under Relaxation Rules and click **Edit**. • Verify there are 3 new rules (5 enabled rules total) in the Start URL list. Click **Close**. URL Closure is now properly configured to allow regular navigation of WebGoat.
14.	Keep the WebGoat profile properties open.

CSRF 3: Test URL Closure without Referer Header Validation Protection

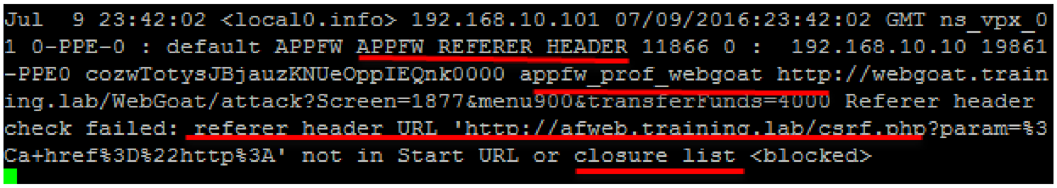
Step	Action

1.	Clear all session state in Firefox before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate method: click **History > Clear Recent History**. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared. • Re-open Firefox.
2.	Open a new tab in Firefox (Tab 1), browse to the CSRF page in WebGoat and verify the current Screen parameter: • Browse to **http://webgoat.training.lab/WebGoat/attack**: ○ Enter credentials **guest / guest**. ○ Click **Start WebGoat**. • Navigate to **Cross-Site Scripting (XSS) > Cross Site Request Forgery (CSRF)**. • Take note of the current value assigned to the Screen parameter in the URL: `http://webgoat.training.lab/WebGoat/attack?Screen=XXXX&menu900` Copy the URL to Notepad or write down the value for Screen. Screen ID:________________________________ This value may have changed from the previous test, if you reset Firefox.
	Open a second Tab (Tab 2) in Firefox, browse to the CSRF Attack page in AFWeb: • Browse to **http://afweb.training.lab/**. • Click on the **CSRF Demo.** Do not perform the attack yet.
3.	Open Live HTTP Headers in a new tab (Tab 3): • Click **Tools > Live HTTP Headers**.

4.	Return to the AFWeb tab (Tab 2), in Firefox and perform the CSRF attack: • Enter the HREF sample into the custom message field. **`<a href="http://webgoat.training.lab/WebGoat/attack?Screen=XXXX&menu900&transferFunds=4000">Safe to Click</a>`** • Replace **XXXX** with the current value for the WebGoat CSRF attack page screen value. • Click **Submit**. • Click **Safe to Click**. The attack is successful. **Explanation**: The attack was not blocked by the Application Firewall. The URL Closure ensured that WebGoat was navigable and all required content loaded. Attacks that result in direct browsing to hidden links will fail. However, the CSRF attack originates from a different website, and URL Closure alone doesn't prevent this attack.
5.	Switch to the Live HTTP Headers (Tab 3): • Scroll to the top of the capture. • The first request in the capture should be the request to http://afweb.training.lab/csrf.php?<query string>. • The second request in the capture should be the request to http://webgoat.training.lab with the TransferFunds string. • Take note of the Referer header in this request. **Result**: The Referer header confirms that the request to the WebGoat page originated from a remote server at http://afweb.training.lab/csrf.php.
6.	Return to the NetScaler configuration utility in Chrome: http://192.168.10.101.
7.	Update the WebGoat profile to include Referer Header validation: • Select (check) **Start URL** under Security Checks and click **Action Settings**. • Select **If Present** under Enable Validate Referer Header. Click **OK** to close the Start URL Settings. Click **Done** to close the profile.
8.	Save the NetScaler configuration.

CSRF 4: Test Referer Header Validation Protection to Prevent the CSRF Attack

Step	Action
1.	Clear all session state in Firefox before continuing: • Make sure the Firefox window has focus, then enter **CTRL+SHIFT+DEL**. Alternate method: click **History > Clear Recent History**. • Ensure **Everything** is selected in the Time Range to Clear box and **Cookies** are included in the Details. • Click **Clear Now**. **IMPORTANT**: • Close all instances of Firefox, then re-open a new window, before continuing. This ensures any session cookies are also cleared. • Re-open Firefox. **Do not** browse to WebGoat or AFWeb yet.
2.	You should be able to repeat the test with the current setup. • If you reset Firefox between attempts, then the Screen ID may change again. • The initial steps will reconfirm the setup before performing the attack. Verify FireFox is setup as follows: On Tab 1, browse to the WebGoat CSRF Page: • Browse to **http://webgoat.training.lab/WebGoat/attack**. o Enter credentials **guest / guest**. o Click **Start WebGoat**. • Navigate to **Cross-Site Scripting (XSS) > Cross Site Request Forgery (CSRF)**. • Click **Restart this Lesson** to reset the state, if needed. • Take note of the current value assigned to the Screen parameter in the URL: **`http://webgoat.training.lab/WebGoat/attack?`** **`Screen=XXXX&menu900`** Screen ID:________________________________
3.	On Tab 2, browse to AFWeb: • Browse to **http://afweb.training.lab**. • Click **CSRF Demo** to load the /csrf.htm page. On Tab 3, run LiveHTTPHeaders: • Click **Tools > Live HTTP Headers**. • Ensure **Capture** is still enabled (checked) at bottom of page. • Click **Clear** to clear the current capture output.

4.	Switch to Tab 2 in Firefox which is already on AFWeb's CSRF demo page (/csrf.htm). Perform the CSRF attack: • Enter the HREF sample into the custom message field. `<a href="http://webgoat.training.lab/WebGoat/attack?Screen=XXXX&menu900&transferFunds=4000">Safe to Click</a>` • Replace **XXXX** with the current value for the WebGoat CSRF attack page screen value. • Click **Submit**. • Click **Safe to Click**. **Result**: This time the attack is blocked by Application Firewall and the WebGoat blocked page is displayed.
5.	Switch to the **Putty (1)** window displaying the syslog (/var/log/ns.log) output: Identify the violation displayed: Confirm a Referer Header violation is logged from the WebGoat profile, instead of a Start URL or other violation. Referrals from http://afweb.training.lab are disallowed as it is not included in the Start URL Closure.  `Jul 9 23:42:02 <local0.info> 192.168.10.101 07/09/2016:23:42:02 GMT ns_vpx_01 0-PPE-0 : default APPFW APPFW_REFERER_HEADER 11866 0 : 192.168.10.10 19861-PPE0 cozwTotysJBjauzKNUeOppIEQnk0000 appfw_prof_webgoat http://webgoat.training.lab/WebGoat/attack?Screen=1877&menu900&transferFunds=4000 Referer header check failed: referer header URL 'http://afweb.training.lab/csrf.php?param=%3Ca+href%3D%22http%3A' not in Start URL or closure list <blocked>` NOTE: If the AFWeb URL is added to the WebGoat Start URL list, then this referral would be valid.
6.	Return to the NetScaler Configuration Utility in Chrome at http://192.168.10.101.
7.	Re-enable Application Firewall protection for AFWeb: • Navigate to **Security > Application Firewall > Policies > Firewall**. • Select (check) **appfw_pol_afweb** and click **Edit**. • Change the expression from false to **true**. Click **OK** to close the policy.
8.	Save the NetScaler configuration.

Takeaways:

- CSRF Form Tagging and Referer Header Validation protection are two types of CSRF attack preventions.
    - CSRF Form Tagging is slightly more CPU intensive than Referer Header validation.
    - CSRF Form Tagging and Referer Header validation requires that Form Tagging is enabled at the profile settings.
    - With CSRF, each response sent to users is tagged with a custom Form ID. Any request submitted to the NetScaler, must contain a valid Form ID or it will be in violation.
- Both CSRF Form Tagging and Referer Header validation depends on the Start URLs being properly configured. Do not enable either security check until Start URLs (with or without Closure) is configured for regular site navigation.

- Referer Header validation requires that requests to a protected Website must originate from a source covered by the Start URLs or otherwise covered by URL Closure.
- CSRF Form Tagging automatically identifies a violation if an action URL is triggered from an origin URL in a different Domain than the action URL (example.com vs. example.net).

Module 5: Application Firewall Logs and Troubleshooting

Overview:

In this module, you will perform hands-on exercises using Application Firewall violations in syslog to aid in troubleshooting. This module will examine creating custom error pages that integrate Application Firewall log details and viewing Security Insight AppFlow data within NetScaler Insight Center.

After completing this lab module, you will be able to:

- View Application Firewall events in syslog and identify the violation that has occurred (based on Module 4 exercises).
- Use Application Firewall events in syslog to update the Application Firewall based on Module 4 exercises.
- Create a custom error page for import onto the NetScaler that includes Application Firewall log event details.
- Integrate Security Insight reporting and interpret the Security Insight Dashboard reports in NetScaler Insight Center.

This module contains the following exercises using the NetScaler Configuration Utility GUI:

• Exercise 5-1: Custom Application Firewall Error Page	10 min
• Exercise 5-2: NetScaler Insight Center: Security Insight	15 min

Before you begin:

Estimated time to complete this lab module: 25 minutes

Exercise 5-1: Custom Application Firewall Error Page

In this exercise, you will import a custom error page for Application Firewall violations that will include Application Firewall log information such as Transaction ID, Application Firewall Session ID, and violation details. The error page is useful for detailed debugging and demonstrates how certain parameters can be retrieved from the NetScaler using variables in the page content.

This information may be useful for providing debug information when users report blocked content. Care should be exercised in its use as the page in its current form clearly identifies the presence of NetScaler Application Firewall in use within the environment. Page content may be appropriate for use by internal users during initial acceptance testing while adjusting the Application Firewall configuration. The page may require modifications to obscure references to the NetScaler Application Firewall prior to being used in production.

In this exercise, you will perform the following tasks:

- Import a custom error page from an existing file that contains variables to report Application Firewall violation details.
- Configure the custom import as a "block" page for violations within an Application Firewall profile.

Configure a Custom Application Firewall Error Page for WebGoat

Step	Action
1.	On the Student Desktop, browse to **C:\resources\**. • Right-click **CustomAppFw.html** and click **Edit with Notepad++**. • View the custom error page. • Close page when done. For reference, the custom HTML Page consists of the following code:

```
<html>

<head>
   <title>Application Firewall Block Page</title>
</head>
<body>
   <h1><B>Your request has been blocked by a security
policy<B><BR></H1>
   <H3>Access has been blocked - if you feel this is in error,
please contact the site administrators quoting the following
details: </H3>
   <UL>
         <li>NS Transaction ID: ${NS_TRANSACTION_ID}
         <li>AppFW Session ID: ${NS_APPFW_SESSION_ID}
         <li>Violation Category:
${NS_APPFW_VIOLATION_CATEGORY}
         <li>Violation Details: ${NS_APPFW_VIOLATION_LOG}
   </UL>
</body>

</html>
```

	IMPORTANT: This page content is useful when used as an application firewall blocked page during limited types of controlled User Acceptance Testing (UAT). It advertises a lot of sensitive information about the Application Firewall, violations, and security settings. This level of information means it is not appropriate as a production blocked page that any malicious user could see. Do not use this type of page content as a blocked page for a production Application Firewall profile.
2.	Return to the NetScaler configuration utility and connect to the NSIP: http://192.168.10.101.
3.	Create a new error page for use with WebGoat: • Navigate to **Security > Application Firewall > Imports**. • Click **Add**. • Enter **adverror_webgoat** in the Name field. • Select **File** under Import From option. • Click **Choose File** to browse: Browse to **C:\resources\** and select **CustomAppFw.html**. Click **Open**. Click **Continue**. View the imported file contents. Click **Done** to close the HTML Error Page Import Object screen. **NOTE**: • The file contents can be edited on the NetScaler after import. • Imported files are located in: /var/download/. • This file will display the Application Firewall violation details found in syslog including the NetScaler Transaction ID, Session ID, Violation Category, and Violation Log message.
4.	Update the error page associated with the WebGoat profile: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **appfw_prof_webgoat** and click **Edit**. • Select **Profile Settings** under Advanced Settings. • Select **adverror_webgoat** under HTML Error Object. Click **OK**.
5.	Enable Form Field Consistency check: • Select **Security Checks** under Advanced Settings. • Enable **Block**, **Log**, and **Stats** next to **Form Field Consistency** under Security Checks. • Click **OK** to apply settings. Click **OK** to confirm Form Tagging has been enabled, if prompted.
6.	Switch to Firefox: • Browse to **http://webgoat.training.lab/WebGoat/attack**. • Navigate to **Parameter Tampering > Bypass HTML Field Restrictions**.

7.	Use Web Developer Extension to perform the attack in WebGoat: • Click **Tools > Web Developer Extension > Forms > Display Form Details.** • Delete contents of field **as_fid**. (This violates the form field consistency validation.) Click **Submit**.
8.	View the Custom Application Firewall error page.
9.	Compare the error page results with the violation in the **Putty (1)** session displaying syslog output. Switch to the Putty window displaying the syslog (/var/log/ns.log) output: Confirm that Syslog displays the same information present in the custom error page: • NetScaler Transaction ID • AppFW Session ID • Violation Category • Violation Details
10.	Return to the NetScaler configuration utility.
11.	Click **Done** to close the WebGoat Profile.
12.	Save NetScaler configuration.

Takeaways:

- On block violations, the Application Firewall can redirect users to the default page ("/"), another server hosting a dedicated error page, or custom content can be uploaded or generated on the NetScaler and delivered from itself.
- Custom error pages can integrate NetScaler variables which allows displaying event ID's and other descriptive information in the error page. This information can be useful for debugging.
- For test purposes, it can be useful to confirm when an Application Firewall violation occurred and the details of that violation. In production, usually it is better to provide an event ID that can be tracked by support teams, but no overt information regarding the security violation prevented or the mechanism used to provide the protection. Be cautious when constructing error pages for production use so that you are not providing information on how to circumvent required security.

Exercise 5-2: NetScaler Insight Center: Security Insight

In this exercise, you will enable Security Insight with NetScaler Insight Center and review the logging information available in Security Insight dashboard.

NetScaler Insight Center integration with the NetScaler was initially configured in Module 2. During this exercise, the final integration will be configured to enable Security Insight data reporting. Once reporting is enabled, additional application attacks will be performed to generate new Application Firewall violation data. The Security Insight data for AFWeb and WebGoat will be reviewed.

The lab will show the basic interactions with the Security Insight dashboard. Students may explore Security Insight data in more detail.

In this exercise, you will perform the following tasks:

- Enable AppFlow Security Insight Traffic reporting on the NetScaler.
- Create new Application Firewall events.
- View Security Insight reports for protection level and Application Firewall violation events.

Enable Insight Security Reporting and View Insight Security Dashboard

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**

2.	Enable additional AppFlow settings to enable Security Insight:
	• Navigate to **System > AppFlow**.
	• Click **Change AppFlow Settings**.
	• Enable **Security Insight Traffic**. This enables Security Insight data gathering.
	• Set Security Insight Record Interval to **60** (seconds). Default is 600 seconds (10 min).
	• Keep all other settings with their current values.
	Click **OK**.
	Note:
	For reference, the command line equivalent for the above parameters are:
	`set appflow param -SecurityInsightTraffic ENABLED`
	`set appflow param -SecurityInsightRecordInterval 60`
	The NetScaler Insight Center "enable Security Insight" wizard does not enable the above settings. Security Insight AppFlow reporting is not enabled until the "Security Insight Traffic" parameter is enabled.

3.	Save the NetScaler configuration.

4.	Generate Application Firewall violations for AFWeb: Switch to Firefox and access AFWeb: • Browse to **http://afweb.training.lab**. • Click on the **Deny Demo** link. • Click on the **Buffer Overflow 2 Demo** link. • Click on the **SQL Injection Demo** link. o Enter **Select '** in the Lookup Value field. o Click **Submit**. • Manually browse directly to **http://afweb.training.lab/private2.htm**.
5.	Generate Application Firewall violations for WebGoat: Open a new tab in Firefox and access WebGoat: • Browse to **http://webgoat.training.lab/WebGoat/attack**. o Log on as **guest / guest**, if prompted. o Click **Start WebGoat** on the intro page if required. • Navigate to **Parameter Tampering > Bypass HTML Field Restrictions**. • Use Web Developer Extension to display hidden fields: o In Firefox, Click **Tools > Web Developer Extension > Forms > Display Form Details**. o Clear the field value for **as_fid** and set it to <blank>. • Click **Submit**.
6.	Switch to Chrome and open a new tab for NetScaler Insight Center: • Browse to http://192.168.10.13. • Log on as **nsroot / nsroot**.

7.

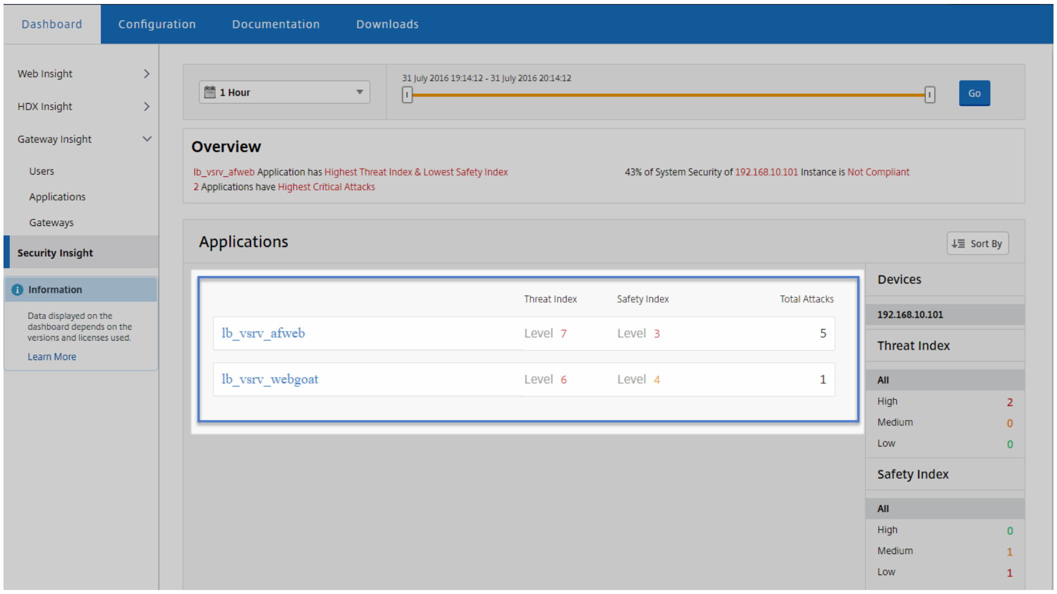
View the Security Insight Dashboard:

- Click **Dashboard** tab.
- Navigate to **Security Insight**.
- Click **Get Started**.

View preliminary Application details:

- Under applications, both **lb_vsrv_afweb** and **lb_vsrv_webgoat** should be listed.
- Attacks should be listed for AFWeb.
  - ○ There may be a few minute delay between the time that attacks are generated and reported to AppFlow. You may have to wait 5-10 minutes for the first data set to be reported.
- Initial Threat and Safety Indexes are displayed. (Your values may vary from image.)

It may take a few minutes before data for both applications have been reported. Times will vary, but it may take up to 10 minutes before data is displayed. Try repeating the attack in steps 4-5 if data still hasn't populated.



Data displayed on the dashboard depends on the versions and licenses used.

31 July 2016 19:14:12 - 31 July 2016 20:14:12

Overview

lb_vsrv_afweb Application has Highest Threat Index & Lowest Safety Index

2 Applications have Highest Critical Attacks

43% of System Security of 192.168.10.101 Instance is Not Compliant

Applications

	Threat Index	Safety Index	Total Attacks
lb_vsrv_afweb	Level 7	Level 3	5
lb_vsrv_webgoat	Level 6	Level 4	1

Devices

192.168.10.101

Threat Index

All	
High	2
Medium	0
Low	0

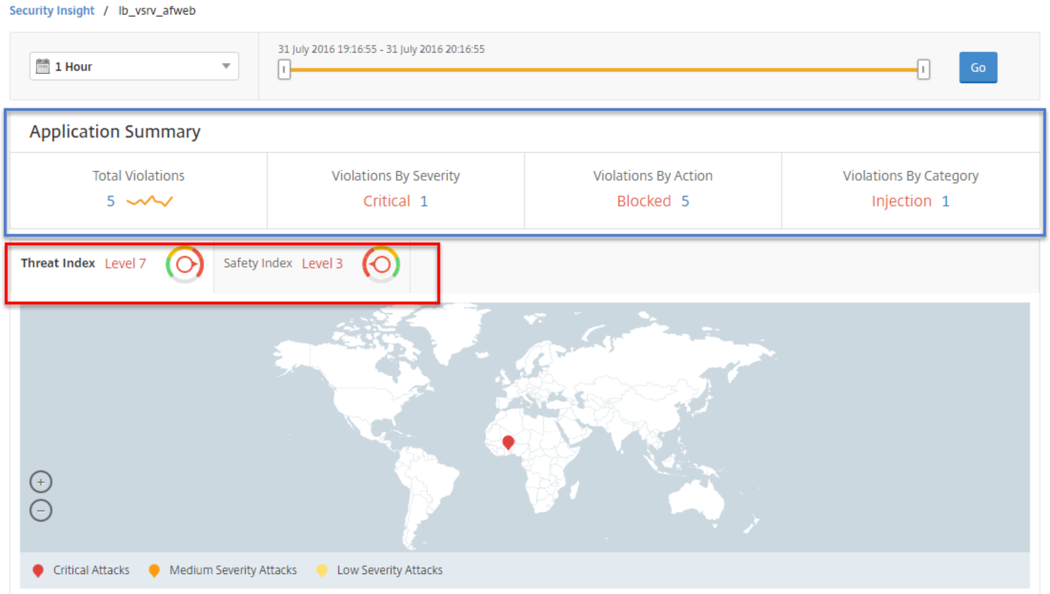
Safety Index

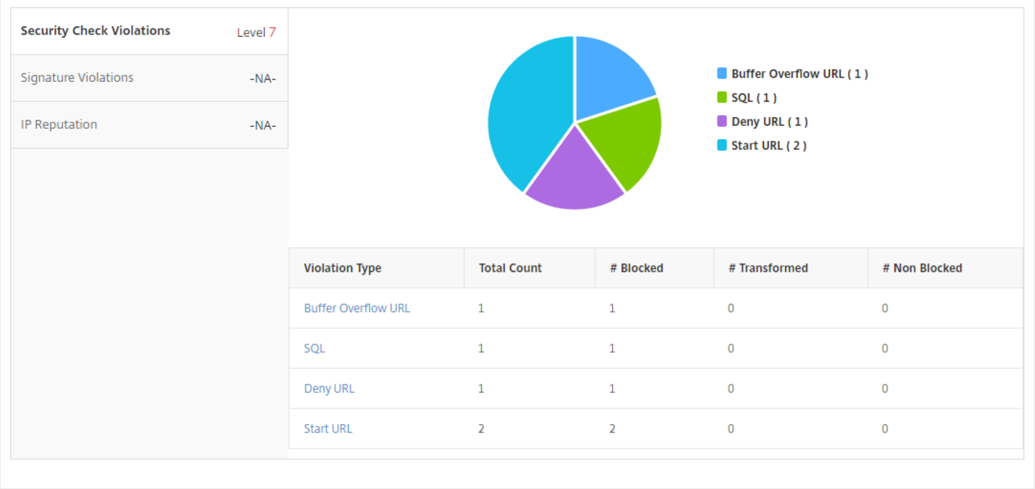
All	
High	0
Medium	1
Low	1

8.

View the Security Insight data for **lb_vsrv_afweb**.

- Click **lb_vsrv_afweb** to display the full details for this virtual server.

9.	Review the Application Summary: • GeoIP reporting is not enabled, so the geographical origins of the attacks are not being reported (and is being affected by the lab IP scheme, as well.) • Notice that in the Summary pane, the following areas are clickable and can be used to drill-down into the attacks: o Total Violations o Violations by Severity o Violations by Action o Violations by Category • The following areas allow you to switch between Threats (views of attacks) vs. Safety (profile configuration/protections applied). o Threat Index o Safety Index • The lower half of the chart, displays the breakdown of the current view. Clicking on an attack violation type, displays the attacks reported for that violation.
	 Security Insight / lb_vsrv_afweb 1 Hour 31 July 2016 19:16:55 - 31 July 2016 20:16:55 Go Application Summary Total Violations 5 Violations By Severity Critical 1 Violations By Action Blocked 5 Violations By Category Injection 1 Threat Index Level 7 Safety Index Level 3 ● Critical Attacks ● Medium Severity Attacks ● Low Severity Attacks



Security Check Violations	Level 7
Signature Violations	-NA-
IP Reputation	-NA-

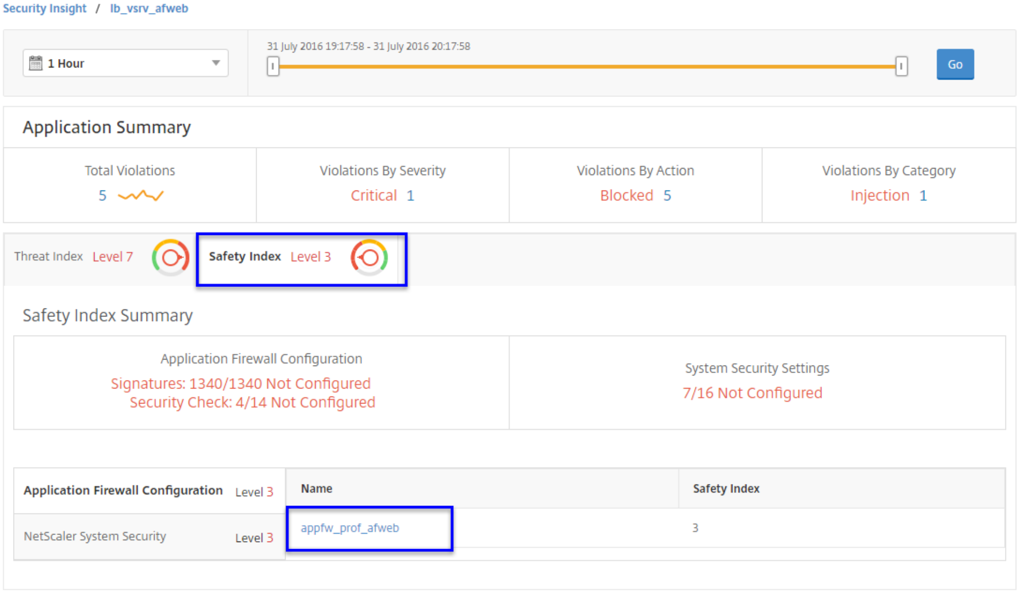
Violation Type	Total Count	# Blocked	# Transformed	# Non Blocked
Buffer Overflow URL	1	1	0	0
SQL	1	1	0	0
Deny URL	1	1	0	0
Start URL	2	2	0	0

Note:

- NetScaler Security Insight only retains minutely data for 2 hours.
- Hourly data persists for 1 Day.
- Daily data persists for 31 Days.

10. Within the Application Summary for lb_vsrv_afweb,

- Click **Safety Index** to bring up the Safety Index Summary.
- Click **appfw_prof_afweb** to drill into the profile configuration summary.



Security Insight / lb_vsrv_afweb

Application Summary

Total Violations	Violations By Severity	Violations By Action	Violations By Category
5	Critical 1	Blocked 5	Injection 1

Threat Index Level 7   Safety Index Level 3

Safety Index Summary

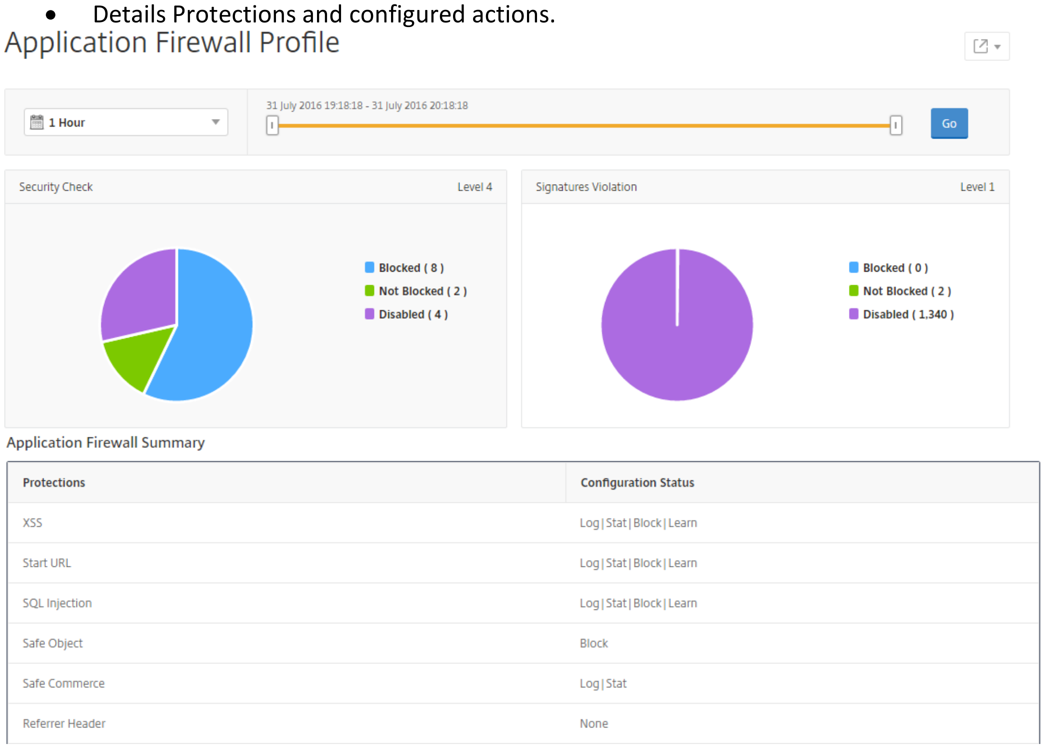
Application Firewall Configuration	System Security Settings
Signatures: 1340/1340 Not Configured Security Check: 4/14 Not Configured	7/16 Not Configured

Application Firewall Configuration	Level 3
NetScaler System Security	Level 3

Name	Safety Index
appfw_prof_afweb	3

11. Review the Application Firewall Profile sumamry for lb_vsrv_afweb:

- Details Protections and configured actions.



Application Firewall Profile

Application Firewall Summary

Protections	Configuration Status
XSS	Log | Stat | Block | Learn
Start URL	Log | Stat | Block | Learn
SQL Injection	Log | Stat | Block | Learn
Safe Object	Block
Safe Commerce	Log | Stat
Referrer Header	None

12. Return to the NetScaler configuration utility in Chrome: http://192.168.10.101.

13. Disable Security Insight Reporting before continuing:

- Navigate to **System > AppFlow**.
- Click **Change AppFlow Settings**.
- Disable (uncheck) **Security Insight Traffic**.
- Click **OK**.

Alternate method, use the CLI in **Putty (2)** to disable Security Insight Traffic:

- Run the following command in the NetScaler CLI:

```
set appflow param -SecurityInsightTraffic DISABLED
```

14. Switch to **Putty (2)** and disable the Application Firewall feature using the NetScaler CLI:

- Run the following command to disable Application Firewall:

```
disable ns feature appfw
```

NOTE: Feature is being disabled to prevent interference with later exercises. Policies are still in place, if students want to revisit Application Firewall testing later.

15. Save the NetScaler configuration.

Takeaways:

- Security Insight uses AppFlow to report Application Firewall violations and statistics to NetScaler Insight Center. It includes reporting for Application Firewall security check violations, signature violations, and IP Reputation.

- The NetScaler Insight Center summarizes violation events and provides an assessment/summary of each integrated applications, threat index and safety index. Threat Index is affected by the number of and type of observed attacks. The safety index is affected by the comprehensiveness of settings integrated and can be used to identify additional security protections that can be applied.

Module 6: Advanced Security and Filtering

Overview:

In this module, you will perform hands-on exercises that will demonstrate other security and traffic filtering capabilities on the NetScaler that can be used in addition to the NetScaler Application Firewall features. These features include HTTP Callouts, IP Rate Limiting, App QOE, and IP Reputation. HTTP Callouts, IP Rate Limiting, and IP Reputation features can be used to define entities that can be incorporated into advanced policies for features such as Responder and Application Firewall, as needed.

After completing this lab module, you will be able to:

- Configure an HTTP Callout expression that can be incorporated into an advanced policy feature, such as a responder policy.
- Configure an IP Rate Limit identifier and selector that can detect request rates per URL that exceed a specific threshold.
- Enable the NetScaler's built-in HTTP challenge-response capabilities and configure AppQOE policy thresholds to trigger the validation process when traffic exceeds certain thresholds.
- Enable IP Reputation and test traffic filtering using this service.

This module contains the following exercises using the NetScaler Configuration Utility GUI:

• Exercise 6-1: HTTP Callouts	25 min
• Exercise 6-2: IP Rate Limiting	20 min
• Exercise 6-3: App QOE	20 min
• Exercise 6-4: IP Reputation	15 min

Before you begin:

Estimated time to complete this lab module: 1 hour 20 minutes

Exercise 6-1: HTTP Callouts

In this exercise, you will construct an HTTP Callout that can pass a client IP Address to a remote callout agent and determine whether the traffic is or is not on the blacklist. The HTTP Callout will be incorporated into a Responder policy to filter unwanted traffic.

Scenario:

As the administrator for the NetScaler, you were asked to integrate a traffic filtering mechanism that will allow the NetScaler to compare IP Addresses of incoming traffic against the internally generated security blacklist.

The blacklist is being maintained on an existing system separate from the NetScaler. The internal blacklist system maintains a database. The blacklist system already has a query mechanism configured via a web page that can be used to retrieve the contents of the database or the web script can evaluate whether a given IP address is found in the database.

The security team would like you to demonstrate the NetScaler filtering capabilities by incorporating a traffic filter on the NetScaler that can compare traffic against the blacklist database and take an appropriate filter action. They can then determine how to apply to a broader range of applications at a later point in time.

Requirements for this scenario:

- Construct an HTTP Callout that can pass an IP Address to the callout agent and identify if the IP Address is or is not on the blacklist.
- Use the HTTP Callout to trigger a responder policy to drop blacklisted IP Addresses.
- Test the policy against the RBG load balancing virtual server.
- At the end of the demonstration, the policy will be unbound.

Callout Details**:**

Entity	Value
Callout Server IP (Backend)	192.168.30.79
Load Balancing vServer IP (lb_vsrv_callout)	172.21.10.119
Callout Script Path	/cgi-bin/check_client.pl

In this exercise, you will perform the following tasks:

- View the HTTP Callout agent and identify how to pass and retrieve data from the agent.
- Configure an HTTP Callout that identifies whether the Client IP Address is or is not on the blacklist by evaluating the callout response
- Configure HTTP Callouts with both a Boolean and Text-based return types to examine different ways an HTTP Callout can be used to process data.
- Configure a responder policy to drop unwanted traffic based on the HTTP Callout evaluation.

Configure HTTP Callouts for IP Blacklist Evaluation

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
2.	Configure NetScaler Basic features: • Navigate to **System > Settings**. • Click **Configure Basic Features**. • Disable (uncheck) **Application Firewall**. • Verify **Rewrite** is enabled. Click **OK**. Configure NetScaler Advanced Features • Click **Configure Advanced Features**. • Verify **Responder** is enabled. Click **OK**.
3.	Switch to Firefox and browse to **http://rbg.training.lab/home.php.** • Verify the content is displayed. Prior to creating the callout and the responder policy to filter traffic, any request can successfully reach the RBG virtual server.
4.	Return to the NetScaler Configuration Utility in Chrome: http://192.168.10.101.
5.	Create a load balancing virtual server for the Callout Service: Create an HTTP Service: • Navigate to **Traffic Management > Load Balancing > Services**. • Click **Add**. • Enter **svc_callout** in the Service Name field. • Enter **192.168.30.79** in the IP Address field. • Verify Protocol is set to HTTP. • Verify Port is set to 80. Click **OK** and click **Done** to create the service.

6.	Create a load balancing virtual server for the callout server: • Navigate to **Traffic Management > Load Balancing > Virtual Servers**. • Click **Add**. • Enter **lb_vsrv_callout** in the Name field. • Verify Protocol is set to HTTP. • Enter **172.21.10.119** in the IP Address field. • Verify Port is set to 80. Click **OK**.
7.	Bind the service the virtual server: • Click **Load Balancing Virtual Server Service Binding** under Services and Service Groups category. • Click **Click to Select** under Select Service. • Select **svc_callout** and click **Select**. • Click **Bind**. • Click **Continue**. Click **Done**. **NOTE**: The callout virtual server (lb_vsrv_callout) may initially appear as DOWN after creation. The in-page refresh should result in it displaying in an UP state.
8.	Test the HTTP Callout agent manually: • Switch to Firefox and browse to: **http://172.21.10.119/cgi-bin/check_client.pl**. View the Callout output. This particular callout generates two types of output. The first just returns the complete IP Address blacklist table, which could allow the NetScaler to evaluate the blacklist itself. The second type of output results when the remote Callout server evaluates the IP Address and determines if the request is or isn't in the blacklist. In this case, the callout returns the phrase IP Failed or IP Matched.
9.	Test the HTTP Callout agent manually with an IP Address in the blacklist: • In a new tab (Tab 2), open Live HTTP Headers in Firefox: Click **Tools > Live HTTP Headers**. ○ If it was already open, click **Clear** to clear the existing output. • Switch to Tab 1, manually browse to the URL: **http://172.21.10.119/cgi-bin/check_client.pl?cip=192.168.10.10**. Verify **IP Matched** is returned, indicating the IP address is on the blacklist. Switch to the Live HTTP Headers tab (Tab 2) and view the GET request associated with this URL and the response headers it generates.
10.	Test the HTTP Callout agent manually with an IP Address not in the blacklist: • Switch to Tab 1, manually browse to the URL: **http://172.21.10.119/cgi-bin/check_client.pl?cip=12.178.35.10**. Verify **IP Failed** is returned indicating the IP address is not on the blacklist.
11.	Return to the NetScaler Configuration Utility in Chrome: http://192.168.10.101.

12. Create an HTTP Callout to the check_client.pl script.

- Navigate to **AppExpert > HTTP Callouts**.
- Click **Add**.
- Enter **hc_blacklist** in the Name field.

Define settings for the callout destination under the Server to receive callout request section:

- Select **Virtual Server**.
- Select **lb_vsrv_callout** under the Virtual Server drop-down list.

13. Define the HTTP Callout web request settings under the Request to send to the server section:

- Select **Attribute-based** under Request Type.
- Select **Get** under Method.
- Enter **"172.21.10.119"** in the Host Expression field. (Quotes required where included.)
- Enter **"/cgi-bin/check_client.pl"** in the URL Stem Expression.

Configure custom Header name-value pairs:

- Click **Insert** under Headers.
- Enter **callout** in the Name field.
- Enter **"negate"** in the Value field.
- Click **Insert**.

Configure custom Query Parameter name-value pairs:

- Click **Insert** under Parameters.
- Enter **cip** in the name field.
- Enter **CLIENT.IP.SRC** in the Value field.
- Click **Insert**.

Verify HTTP Request Scheme:

- Select **http** under Scheme.

NOTE: These parameters define the HTTP request to generate when performing the Callout. The scheme (HTTP or HTTPS) needs to correspond with the destination callout protocol, whether it is a LB virtual server or a direct IP destination. If the virtual server is SSL, be sure the scheme is specified as HTTPS.

14. Define the HTTP Callout web response settings under the Server Response section:

- Select **BOOL** under Return Type.
- Enter the following expression in the Expression field to extract data from the response body (or use the Expression Editor to build the expression):

```
HTTP.RES.BODY(1000).CONTAINS("IP Matched")
```

Click **Create**.

NOTE: This particular callout evaluates whether the remote agent determined if the IP address was or wasn't on the blacklist and outputs a Boolean result TRUE or FALSE. Comparison is case-sensitive.

15.

Create a second HTTP Callout with slightly modified settings, based on the existing callout:

- Select (check) **hc_blacklist** in the HTTP Callouts list and click **Add**. This will create a duplicate callout based on the existing settings.
- Enter **hc_blacklist2** in the Name field.
- Scroll down to the Server Response section.
- Select **Text** under Return Type.
- Enter the following expression in the Expression to extract data from the response field:

```
HTTP.RES.BODY(1000)
```

Click **Create**.

NOTE: This callout returns the list of IPs from the callout server, allowing the NetScaler itself to evaluate if an IP Address is or isn't on the blacklist at the time the callout is used. This particular callout could be cached, as the blacklist is unlikely to change frequently.

This example is used to demonstrate how the response expression and the callout data type are related.

16.

Create a responder policy to DROP content, if a client IP is on the blacklist:

- Navigate to **AppExpert > Responder > Policies**.
- Click **Add**.
- Enter **rs_pol_drop_bycallout** in the Name field.
- Select **DROP** under Action.
- Enter the following expression in the Expression field or use the Expression Editor:

```
SYS.HTTP_CALLOUT(hc_blacklist)
```

IMPORTANT: Choose HTTP_CALLOUT(http_callout_bool) data type when constructing the expression in the Expression Editor.
The Expression Editor will provide a list of callouts with boolean return types. Selecting HTTP_CALLOUT(http_callout_text) will give you a list of callouts with Text return types.

Click **Create**.

17. Create a second responder policy, using the alternate Callout (hc_blacklist2):

- Deselect (Uncheck) **rs_pol_drop_bycallout**.
- Click **Add** (to add a new, blank callout).
- Enter **rs_pol_drop_bycallout2** in the Name field.
- Select **Drop** under Action.
- Enter the following expression in the Expression field or use the Expression Editor:

```
SYS.HTTP_CALLOUT(hc_blacklist2).CONTAINS("IP Matched")
```

IMPORTANT: Choose HTTP_CALLOUT(http_callout_text) data type when constructing the expression in the Expression Editor. This will allow the syntax builder to provide you with text data type operators like Contains().
The Inline-editor will not give the syntax prompts for Contains(), but the expression can still be manually typed in. Use the Expression Editor to build the expression if you have issues with the policy syntax.

Click **Create**.

This policy demonstrates how to create an expression that evaluates the results of the Callout when the callout returns a text-based output.

18. Bind the responder policy to the RBG virtual server, and verify traffic is blocked:

Use Policy manager to select the bind point:

- Remain under Responder > Policies.
- Click **Policy Manager**.
- Select **Load Balancing Virtual Server** under Bind Point.
- Verify HTTP is selected under Protocol.
- Select **lb_vsrv_rbg** under Virtual Server.

Click **Continue**.

Bind the policy:

- Click **Click to Select** under Select Policy.
- Select **rs_pol_drop_bycallout** and click **Select**.
- Verify Priority is set to 100.

Click **Bind**.

Click **Done**.

19. Switch to Firefox and browse to **http://rbg.training.lab/home.php.**

- Verify the content is dropped.

20. Disable the Callout service to simulate a failure:

- Navigate to **Traffic Management > Load Balancing > Services**.
- Select **svc_callout** and click **Action > Disable**.
- Click **OK** to disable service.

21. Verify the Callout state:

- Navigate to **AppExpert > HTTP Callouts**.
- Notice that the Callouts are down since their destination virtual servers are down.

22.	Switch to Firefox and browse to **http://rbg.training.lab/home.php.** • Verify the content is permitted. **NOTE**: • If a callout destination server or virtual server is inaccessible the Callout is down and is not invoked. The policy referencing the callout receives an automatic false, which usually means the policy does not apply. • When configuring HTTP Callouts, use virtual servers and be sure to include backup virtual servers to ensure continued availability of the HTTP Callout function. Leave the callout service disabled before proceeding to later exercises.
23.	Unbind the policy from lb_vsrv_rbg to avoid conflicts with later exercises: Use Policy Manager to select the bind point: • Navigate to **AppExpert > Responder > Policies**. • Click **Policy Manager**. • Select **Load Balancing Virtual Server** under Bind Point. • Verify HTTP is selected under Protocol. • Select **lb_vsrv_rbg** under Virtual Server. Click **Continue**. Unbind the Policy: • Select (check) **rs_pol_drop_bycallout**. • Click **Unbind**. • Click **Yes** to confirm. Click **Done**.
24.	Save the NetScaler configuration.

Takeaways:

- HTTP Callouts can be used to make HTTP or HTTPS requests against a remote agent and retrieve and parse the web response.
- While HTTP Callouts are often associated with filtering traffic as part of Responder or Application Firewall policies, HTTP Callouts can be used with other default policy engine features, including Rewrite and token-based load balancing.
- The HTTP Callout consists of three types of settings: the traffic destination by IP Address or virtual server, the HTTP Request, and the output to evaluate in the HTTP Response. How the HTTP response is processed determines the output return type of the HTTP Callout.
- If the destination for the HTTP Callouts is down, the HTTP Callout does not get invoked and returns an automatic "False" value. This could apply whether the callout points to a direct IP Address destination or a virtual server. In most cases, this will result in the policy where the Callout is referenced not being triggered. In most situations, it is therefore recommended to point the HTTP Callout to a load balancing virtual server with multiple bound services or a backup entity specified to avoid a single point of failure.

Exercise 6-2: IP Rate Limiting

In this exercise, you will create an IP Rate Limit that will trigger a high threshold based on a request rate per URL. The IP Rate Limit will be incorporated into a Responder policy that will redirect traffic to an error page for requests that exceed the limit threshold.

Scenario:

After the HTTP Callout demonstration, one of the security administrators wants to examine other ways of selectively filtering unwanted traffic based on request rates. The security administrator wants you to limit the number of requests per second that can be sent to a given application and wants the requests limit to be tracked per URL.

For demonstration purposes, the request rate threshold will be set artificially low.

Requirements for this Scenario:

- Configure a rate limit that will track requests per URL with a threshold of 5 requests per 10 second interval.
- Upon violation of the request rate, use a responder policy to present a custom error message indicating the request rate as the issue.
- Test the rate limit condition using the WebGoat load balancing virtual server, while Application Firewall is disabled.
- At the end of the exercise, the policy will be unbound.

In this exercise, you will perform the following tasks:

- Create a Limit Selector and Limit Identifier with the thresholds specified.
- Import an HTML page for use with responder.
- Configure the IP Rate Limit to trigger a responder redirect policy.

IP Rate Limit

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
2.	Create a Rate Limit Selector: • Navigate to **AppExpert > Rate Limiting > Selectors**. • Click **Add**. • Enter **limit_sel_byurl** in the Name field. • Click **Insert**. • Enter the following expression in the Expression field: `HTTP.REQ.URL.PATH` • Click **Insert**. Click **Create**.

3.	Create a Rate Limit Identifier that places a limit on the request rate observed per URL. • Navigate to **AppExpert > Rate Limiting > Limit Identifiers**. • Click **Add**. Enter the following parameters for the Limit Identifier: • Name: **limit_highreqrate_byurl** • Selector: **limit_sel_byurl**. • Mode: **REQUEST_RATE**. • Limit Type: **BURSTY**. • Threshold: **5**. • Time Slice (msec): **10000** (10 seconds) Click **Create**.
4.	Import an HTML Page for use with Responder Policy Actions: • Navigate to **AppExpert > Responder > HTML Page Imports**. • Click **Add**. • Enter **ErrorRateLimit** in the Name field. • Select **File** under Import From. • Click **Choose File** to browse the local file system. • Select **C:\resources\ErrorRateLimit.html**. • Click **Open**. • Click **Continue**. Verify the HTML content as is or edit. Click **Done** to import the default page. **NOTE**: Responder policy imports are separate from Application Firewall imports as far as object types go. • Responder imports are managed in the GUI under AppExpert > Responder > HTML Page Imports and in the CLI as "[show | import] responder htmlpage". • Application Firewall imports are managed in the GUI under Security > Application Firewall > Imports and in the CLI as "[show | import] appfw htmlerrorpage". • All imports for both features are stored /var/download/.
5.	Create a Responder Policy Action to respond with an imported page: • Navigate to **AppExpert > Responder > Actions**. • Click **Add**. • Enter **rs_act_respondwith_err_reqrate** in the Name field. • Select **Respond with HTML Page** under Type. • Select **ErrorRateLimit** from the HTML Page drop-down list. Click **Create**.

6. Create a Responder Policy that is triggered by exceeding the rate limit:

- Navigate to **AppExpert > Responder > Policies**.
- Click **Add**.
- Enter **rs_pol_respondwith_err_reqrate** in the Name field.
- Select **rs_act_respondwith_err_reqrate** in the Action field.
- Enter the following expression in the Expression field. Use the inline editor or the expression editor to construct the expression.

```
http.req.url.contains("/WebGoat/") &&
SYS.CHECK_LIMIT("limit_highreqrate_byurl")
```

Click **Create**.

NOTE: Quotes are required around the limit identifier name in the SYS.CHECK_LIMIT() operator. NetScaler 11.0 omitted these quotes when using the Expression Editor. NetScaler 11.1 includes them when using the Expression Editor. However the inline editor does not include them. You will receive a syntax error if the quotes are omitted.

7. Bind the policy to the WebGoat load balancing virtual server.

Select the Bind Point:

- Click **Policy manager**.
- Select **Load Balancing Virtual Server** under Bind Point.
- Verify Protocol is set to HTTP.
- Select **lb_vsrv_webgoat** under Virtual Server.

Click **Continue**.

Bind the policy:

- Click **Click to Select** under Select Policy.
- Select **rs_pol_respondwith_err_reqrate** and click **Select**.
- Verify Priority is set to 100.

Click **Bind**.

Click **Done**.

8. Switch to Firefox and browse to **http://webgoat.training.lab/WebGoat/attack.**

- Reload the page multiple times quickly to trigger the redirect.
- Use the bookmark button for WebGoat v5.4 in the Bookmarks toolbar to rapidly send multiple requests. (Using the browser refresh will generate a resend data prompt.)

NOTE: The page will generate errors before the action is applied due to the low threshold specified and the number of requests in the initial page load.

9. Return to the NetScaler Configuration Utility in Chrome: http://192.168.10.101.

10.	View the policy hits and stats for the Rate Limit Identifier: View the Responder policy hits: • Navigate to **AppExpert > Responder > Policies**. • View the number of Policy hits that occurred for rs_pol_respondwith_err_reqrate (indicating the number of times the threshold was exceeded.) Scroll right or use the Statistics option to view policy hits. View the Rate Limit Identifier Stats: • Navigate to **AppExpert > Rate Limiting > Limit Identifiers**. • View the hits on the limit identifier limit_highreqrate_byurl. • View the Action Taken reported indicating the number of times the threshold was exceeded. Remain on the Limit Identifiers node.
11.	Switch to Firefox and refresh the page one or twice (not enough to exceed the threshold): **http://webgoat.training.lab/WebGoat/attack**.
12.	Return to the NetScaler Configuration Utility in Chrome: http://192.168.10.101.
13.	View the sessions tracked by the limit identifier: • Select (check) **limit_highreqrate_byurl** and click **Action > Show Sessions**. • View how the limit identifier tracks hits per URL (based on the limit selector). Click **Close**.
14.	Unbind the policy to restore normal operation. Select the Bind Point: • Navigate to **AppExpert > Responder > Policies**. • Click **Policy Manager**. • Select **Load Balancing Virtual Server** under Bind Point. • Verify Protocol is set to HTTP. • Select **lb_vsrv_webgoat** under Virtual Server. Click **Continue**. Unbind the policy: • Select (check) **rs_pol_respondwith_err_reqrate** and click **Unbind**. • Click **Yes** to confirm. Click **Done**.
15.	Save the NetScaler configuration.

Takeaways:

- The Rate Limiting feature is another protection feature of the NetScaler that can be incorporated into default-policy engine features, like Responder, Application Firewall, DNS, rewrite, and Integrated Caching. The feature is useful when filtering unwanted traffic exceeding certain connection, request rate, or bandwidth thresholds during high-volume traffic events associated with an attack.
- Selectors are optional filters that can be incorporated with Rate Limit Identifiers. The Selectors provide a filter that can be used to categorize traffic. The Limit Identifier is then used to specify the threshold of interest per category of traffic identified by the selector.
- The Limit Identifier returns true when the threshold is exceeded. This allows the Limit Identifier to trigger the required action in the policy it is associated with. Rate Limiting is usually used to drop, reset, or redirect high threshold traffic exceeding the limit identifier.

Exercise 6-3: App QOE

In this exercise, you will configure an AppQOE policy to trigger a user request validation in the form of an HTTP challenge response to identify legitimate traffic under high traffic load conditions.

AppQOE integrates multiple policy-based security features into one integrated feature. This feature provides an updated mechanism for managing protections previously handled by Priority Queuing, HTTP Denial of Service protection, and SureConnect. AppQOE also integrates a fair queuing system that works at the virtual server level instead of at the service level, allowing traffic handling to occur prior to load balancing instead of afterwards. This exercise will demonstrate denial of service protection using the NetScaler's built-in challenge response/Captcha feature.

Scenario:

The security team has, once again, asked you to assist them with a problem. The security team has asked you to help them simulate a high traffic load condition to see how the App QOE feature can be used to protect a specific application with an integrated Captcha response. To simulate high volume traffic a load generation script will be used.

Requirements for this scenario:

- Configure AppQOE parameters and policy to protect the RBG web application.
- Unbind the policy at the end of the demonstration.

In this exercise, you will perform the following tasks:

- Enable the built-in captcha capabilities on the NetScaler.
- Configure AppQOE policies to trigger a Captcha verification as an alternate response after exceeding the connection threshold.
- Use a load generation script to create excess load on the NetScaler.
- Verify the Captcha response is triggered by the AppQOE policy.

Configure App QOE for DOS Protection with NetScaler Integrated CAPTCHA

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
2.	Enable the AppQoE Feature: • Navigate to **System > Settings** • Click **Configure Advanced Features**. • Enable (check) **AppQoE**. Click **OK**.

3. Configure AppQoE Parameters:

- Navigate to **AppExpert > AppQoE**.
- Click **Configure AppQoE Parameters** in the right-pane.

Adjust Parameters for DOS attack Thresholds:

- Session Life (secs): **1**
- Average Waiting Client: **1000000** (default)
- Alternate Response Bandwidth Limit (Mbps): **1**
- DOS Attack Threshold: **0**

Click **OK**.

NOTE: These thresholds are being set low for demonstration purposes in the lab environment. These thresholds may not be suitable for production traffic volumes and may result in premature AppQOE impacts on legitimate traffic volumes.

4. Create an AppQoE action:

- Navigate to **AppExpert > AppQoE > Actions**.
- Click **Add**.

Configure the AppQoE Action.

- Name: **qoe_dos**
- Action Type: **NS**
- Priority: **Lowest**
- Policy Queue Depth: **10**
- Queue Depth: **5**
- Maximum Connections: **2**
- Delay (microseconds): **100**
- DOS Action: **HICResponse**

Click **Create**.

5. Create an AppQoE policy:

- Navigate to **AppExpert > AppQoE > Policies**.
- Click **Add**.

Configure the AppQoE Policy:

- Enter **qoe_pol_dos** in the Name field.
- Select **qoe_dos** in the Action field.
- Enter the following Expression in the Expression field:

```
http.req.is_valid
```

Click **Create**.

6.	Bind the AppQoE policy to the RBG load balancing virtual server: - Navigate to **Traffic Management > Load Balancing > Virtual Servers**. - Select **lb_vsrv_rbg** and click **Edit**. Select the Policy Type to bind to the virtual server: - Click **Policies** under Advanced Settings. - Click **"+" (Add)** next to Policies. - Select **App QOE** under Choose Policy. - Verify Request is selected under Choose Type. - Click **Continue**. Bind the policy: - Click **Click to Select** under Select Policy. - Select **qoe_pol_dos** and click **Select**. - Keep priority set to 100. - Click **Bind**. Click **Done** to close the virtual server properties.

7.	Enable Captcha Integration from the NetScaler CLI:
	- Open Putty and connect to the NSIP address: Right click on **Start > Run > putty 192.168.10.101**
	- Log on as **nsroot / nsroot**.
	- Run the following command from the CLI to access BSD Shell:
	`shell`
	- Run the following nsapimgr command to enable Captcha support:
	`nsapimgr -ys appqos_captcha=1`
	- Verify the command returns a positive confirmation (or else the command syntax was wrong). The correct return message is:
	`Changing cfg_appqoe_captcha_enable from 0 to 1 ….Done.`
	
	NOTE:
	- This command is available from BSD shell only.
	- The nsapimgr commands applied at shell are not retained after a reboot, as these are applied to RAM but not part of the running config and they are not preserved in the save config.
	- To make nsapimgr commands persistent, they must be added to the /nsconfig/rc.netscaler file (See next step).

8. **OPTIONAL**: Append nsapimgr command to rc.netscaler file (to preserve between reboots)

Continue in shell with the following commands.
Navigate to the /nsconfig directory:

```
cd /nsconfig
```

View the contents of the directory and verify rc.netscaler file is present:

```
ls
```

View the contents of rc.netscaler

```
more rc.netscaler
```

Some existing commands are present, therefore any additional commands need to be appended to the file. Run the following command to preserve the nsapimgr settings and append the new command to the end of the file:

```
echo "nsapimgr -ys appqos_captcha=1" >> rc.netscaler
```

Verify the contents of the updated rc.netscaler file:

```
more rc.netscaler
```

For Reference, the final file should contain the following lines:

```
/bin/sh /etc/ntpd_ctl full_start
nsapimgr -ys appqos_captcha=1 >> rc.netscaler
```

9. Save the NetScaler configuration.

- From shell, exit to the CLI:

```
exit
```

- Save the configuration:

```
save ns config
```

10. Switch to Firefox

- Browse to http://rbg.training.lab/home.php
- Verify page is accessible.

11. ABOUT Running the PYTHON script to generate load.

==

DO NOT direct the script to ANY OTHER WEBSITE than the test site in the lab environment.
DO NOT attempt to use the script on ANY public website.

==

IMPORTANT: The Python script that is used to generate load DOES NOT stop running, even after sending the usual CTRL+C command. As a result, we will run two elevated CMD prompts side-by-side. One will be used to execute the script, the other will be used to terminate the Python engine. In this way, you can start and stop the script as needed.

A text file with both commands is located in C:\resources\hulk_commands.txt, so you can copy-and-paste the commands to the appropriate CMD prompts.

Prepare to run the script by setting up the following commands. Do not run the scripts yet.

Open CMD Prompt (1) to run the hulk.py Python script:

- Open an elevated CMD prompt on the Student Desktop. Use the CMD prompt pinned to the taskbar.
- In CMD Prompt (1) run the following command:

```
c:\hulk.py http://rbg.training.lab/home.php
```

Open CMD Prompt (2) to run the taskkill command:

- Right-click the **CMD** prompt (elevated) pinned to the taskbar and click **Command Prompt** to open a second window.
- Run the following command to stop the hulk.py Python script from executing.

```
taskkill /im Python.exe /F
```

During the test, keep both CMD prompts running with access to the commands during the next several steps and you can start/restart the Python script and terminate the process as needed. When the test is done, remember the following:

- If you have issues terminating the script, use task manager to terminate any running python processes.
- Be sure the hulk.py process is stopped before continuing after this exercise.

12. Open Firefox and browse to **http://rbg.training.lab/home.php**

- Use the browser Refresh button to refresh a couple of times and verify the page is working normally.

This should succeed as the load script is not yet running.

13.	Perform the load generation with AppQOE test:
	• In CMD Prompt (1), start the hulk.py command:
	`c:\hulk.py http://rbg.training.lab/home.php`
	• Switch to Firefox and refresh **http://rbg.training.lab/home.php** a few times.
	• Wait for the Captcha prompt to be generated on the NetScaler.
	• Complete the Captcha response and confirm you are directed successfully to the page content.
	o In a few cases, you will not receive the RBG content as the server is under extreme load. If it does not redirect after a few seconds of completing the captcha, stop the load script anyway.
	When you are done with testing, stop the hulk.py Python script from running:
	• In CMD Prompt (2), stop the Python engine:
	`taskkill /im Python.exe /F`
	• Verify the hulk.py output terminated in CMD Prompt (1).

14.	Return to the NetScaler Configuration utility in Chrome: http://192.168.10.101.
15.	Unbind the AppQOE policy: • Navigate to **Traffic Management > Load Balancing > Virtual Servers**. • Select **lb_vsrv_rbg** and click **Edit**. • Click **App QOE Policy** under the Policies category. • Select (check) **qoe_pol_dos** and click **Unbind**. • Click **Yes** to confirm the Unbind action. • Click **Close** to close the Policy Binding. Click **Done** to close the virtual server properties.
16.	Reset the AppQOE Parameters to the original values: • Navigate to **AppExpert > AppQoE**. • Click **Configure AppQoE Parameters** in the right-pane. Configure the following values: • Session Life (secs): **300** • Alternate Response Bandwidth Limit (Mbps): **100** • DOS Attack Threshold: **2000** Click **OK**.
17.	Save the NetScaler Configuration.
18.	Close the CMD Prompts (1) and (2) on the Student Desktop when done with testing.

Takeaways:

- AppQOE integrates multiple policy-based security features into one integrated feature. This feature provides an updated mechanism for managing protections previously handled by Priority Queuing, HTTP Denial of Service protection, and SureConnect.
- AppQOE can provide traffic prioritization and queuing functions. AppQOE policies can be used to identify traffic and prioritize traffic into higher priority queues or lower priority queues.

Exercise 6-4: IP Reputation

In this exercise, you will enable IP Reputation for traffic filtering based on the IP Reputation database of malicious addresses maintained by WebRoot. IP Reputation reports on IP Addresses and provides additional metadata regarding the threat category of the IP Address.

Once enabled, the NetScaler downloads the database from WebRoot. Data is checked for updates once every five minutes. Delta changes will be downloaded when updates are available.

IP Reputation expressions can then be incorporated into Application Firewall, Responder, and other advanced policy engine features to trigger drop/reset or other filter actions. Traffic can then be identified as malicious if it falls into any identified threat category in the IP Reputation database or the traffic can be filtered if it belongs to a specific threat category (Spam, Botnets, Scanners, or other…).

IP Reputation can be used in place of an internally maintained blacklist or in conjunction with internally maintained whitelist and blacklist systems.

Scenario:

After the HTTP Callout demonstration, the security administrator was curious about integrating IP Reputation checks from the WebRoot service with the NetScaler traffic filtering capabilities. At the moment, the security administrator just wants to see how to enable IP Reputation and incorporate the IP Reputation check into the NetScaler for traffic filtering purposes in place of the previous HTTP Callout blacklist example. The security administrator will look at whether to combine the solutions later.

Requirements for this scenario:

- Enable IP Reputation and verify connectivity by checking the IP Reputation log.
- Integrate an IP Reputation check into an Application Firewall policy that will filter unwanted traffic before other Application Firewall policies with full profile protections are processed.
- For this demonstration, the Application Firewall will display a custom error page to indicate that the violation was related to the IP Reputation failure. Later the action can be switched to drop.

In this exercise, you will perform the following tasks:

- Enable IP Reputation on the NetScaler and verify IP Reputation initialization in the IPrep log.
- Construct an Application Firewall block policy that will drop traffic that fails an IP Reputation check and bind the policy so that IP Reputation traffic is dropped prior to full Application Firewall protections are processed.

Configure IP Reputation

Step	Action
1.	Connect to the NetScaler configuration utility for NS_VPX_01 using the NSIP at http://192.168.10.101. Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**
2.	Enable the Application Firewall feature: • Navigate to **System > Settings**. • Click **Configure Basic Features**. • Enable (check) **Application Firewall**. Click **OK**. Enable the IP Reputation feature: • Navigate to **System > Settings**. • Click **Configure Advanced Features**. • Enable (check) **Reputation**. Click **OK**. NOTE: It will take up to 5 minutes before IP Reputation has connected to the BrightCloud service and downloaded the database.
3.	Open a new PuTTY session or use an existing one: • Open PuTTY and Connect to the NetScaler NS_VPX_01 at 192.168.10.101: `Right Click on Start > Run > putty 192.168.10.101` • Log on as **nsroot / nsroot**.

4.

From the NetScaler CLI, verify the NetScaler can ping the WebRoot BrightCloud IP Reputation service:

- Ping the BrightCloud service to verify name resolution:
  ```
  ping api.bcss.brightcloud.com
  ```
  NOTE: The ping will not return an ICMP response, but it will confirm if the NetScaler can resolve the name to an IP Address.
- Enter **CTRL+C** to stop the ping attempt.

NOTE: In the lab environment we are using a wildcard certificate so we can resolve api.bcss.brightcloud.com correctly however, if you need to setup a proxy for IP reputation you must Connect Method and use **ping api.bcti.brightcloud.com** (which is the correct FQDN). Currently even citrix documentation state the incorrect information.

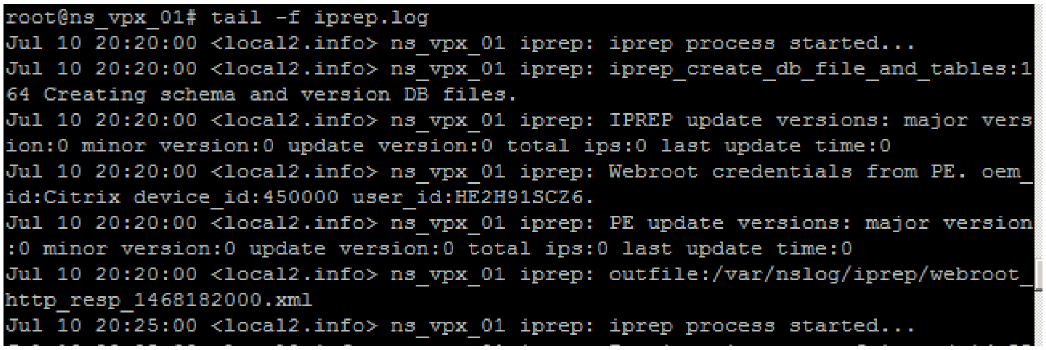
View the IP Reputation log:

- Run the following command to access BSD shell:
  ```
  shell
  ```
- View the /var/log directory:
  ```
  cd /var/log/

  ls
  ```
- Run the following command to view the IPReputation log output as it occurs:
  ```
  tail -f /var/log/iprep.log
  ```

Verify the log confirms the following:

- The IP Reputation (iprep) process started.
- The Database was created
- The NetScaler successfully connected to Webroot (and no authentication errors are listed.)
- The IPRep process performs updates every 5 minutes. It may take 5 minutes after the feature is initially configured before the feature can be used.
- When the update completes, you may see a line saying, "iprep process existing with error code:0". This is a success state after an update completes without error.



```
root@ns_vpx_01# tail -f iprep.log
Jul 10 20:20:00 <local2.info> ns_vpx_01 iprep: iprep process started...
Jul 10 20:20:00 <local2.info> ns_vpx_01 iprep: iprep_create_db_file_and_tables:1
64 Creating schema and version DB files.
Jul 10 20:20:00 <local2.info> ns_vpx_01 iprep: IPREP update versions: major vers
ion:0 minor version:0 update version:0 total ips:0 last update time:0
Jul 10 20:20:00 <local2.info> ns_vpx_01 iprep: Webroot credentials from PE. oem_
id:Citrix device_id:450000 user_id:HE2H91SCZ6.
Jul 10 20:20:00 <local2.info> ns_vpx_01 iprep: PE update versions: major version
:0 minor version:0 update version:0 total ips:0 last update time:0
Jul 10 20:20:00 <local2.info> ns_vpx_01 iprep: outfile:/var/nslog/iprep/webroot_
http_resp_1468182000.xml
Jul 10 20:25:00 <local2.info> ns_vpx_01 iprep: iprep process started...
```

5.	Import a custom Error Page for traffic blocked by IP Reputation failures: • Navigate to **Security > Application Firewall > Imports**. • Click **Add**. • Enter **ErrorIPReputation** in the Name field. • Select **File** under Import From. • Click **Choose File**. • Browse to **C:\Resources\** and select **ErrorIPReputation.html** and click **Open**. • Click **Continue**. Click **Done**.
6.	Edit the built-in APPFW_BLOCK action to display the IP Reputation error: • Navigate to **Security > Application Firewall > Profiles**. • Select (check) **APPFW_BLOCK** and click **Edit**. • Click **Edit** (pencil icon) next to the General category to edit Profile settings. • Select **HTML Error Object** under HTML Error. • Select **erroripreputation** from the HTML Error Object drop down list. • Click **OK** to apply the settings. Click **Done** to close the Profile properties.
7.	Create an Application Firewall Policy that can block traffic based on IP Reputation: • Navigate to **Security > Application Firewall > Policies > Firewall**. • Click **Add**. • Enter **appfw_pol_ipreputation** in the Name field. • Select **APPFW_BLOCK** from the Profile drop-down list. • Configure the Expression to evaluate the IP Reputation. Use the inline editor or the Expression Editor to build the expression: **`CLIENT.IP.SRC.IPREP_IS_MALICIOUS`** Click **Create**. **NOTE**: Other expressions can be used using the IPREP_IS_MALICIOUS or the IPREP_THREAT_CATEGORY(<category>) operators. If retrieving the Source IP from a Layer 7 header (like x-forwarded-for), then the IP Address can be retrieved from the header and typecast to an IP Address, such as: http.REQ.HEADER("x-forwarded-for").TYPECAST_IP_ADDRESS_T.IPREP_IS_MALICIOUS

8.	Bind the policy to the AFWeb load balancing virtual server (at a higher priority than the previous Application Firewall policy). Select the bind point: • Click **Policy Manager**. • Select **Load Balancing Virtual Server** under Bind Point. • Select **lb_vsrv_afweb** under Virtual Server. • Click **Continue**. Bind the policy: • Click **Add Binding**. • Click **Click to Select** under Select Policy. • Select **appfw_pol_ipreputation** and click **Select**. • Enter **10** in the Priority field. • Click **Bind**. Click **Done** to close the Policy Manager for lb_vsrv_afweb.
9.	Open Firefox: • Browse to **http://afweb.training.lab/**. • Verify the content is not blocked as your student desktop should not be on any IP Reputation blacklists.
10.	Return to the NetScaler configuration utility in Chrome at http://192.168.10.101.
11.	View the policy hits for Application Firewall Policies: • Navigate to **Security > Application Firewall > Policies > Firewall**. • If necessary, refresh the configuration view. • Verify that the only policy hits are for the existing appfw_pol_afweb and that the appfw_pol_ipreputation has not applied, meaning your Student Desktop is not on the list of malicious IP Addresses.
12.	Change the policy to temporarily treat your Student Desktop as a malicious IP address: • Select **appfw_pol_ipreputation** and click **Edit**. • Update the expression to trigger if your source IP is NOT malicious. Include the negation operator (!). Enter the following expression: `!CLIENT.IP.SRC.IPREP_IS_MALICIOUS` Click **OK**.
13.	Switch to Firefox, to repeat the test. • Browse to **http://afweb.training.lab/**. • Verify the content is blocked and this time the IP Reputation error page is displayed. **NOTE**: The IP Reputation Error page is being used for demonstration purposes. Normally, traffic failing an IP Reputation test would be dropped using the APPFW_DROP profile.
14.	Return to the NetScaler configuration utility in Chrome at http://192.168.10.101.
15.	View the policy hits for Application Firewall Policies: • Navigate to **Security > Application Firewall > Policies > Firewall**. • If necessary, refresh the configuration view. • Verify this time the appfw_pol_ipreputation policy displays hits.

16.	Restore the policy to expression to the regular state: • Select (check) **appfw_pol_ipreputation** and click **Edit**. • Update the expression to its normal state: Enter the following expression: `CLIENT.IP.SRC.IPREP_IS_MALICIOUS` Click **OK**.
17.	Save the NetScaler configuration.

Takeaways:

- IP Reputation allows the NetScaler to download a list of malicious IP addresses and their threat categorization from list maintained by WebRoot.
- Once enabled, the NetScaler checks for and downloads updates to the database every 5 minutes.
- The NetScaler can then compare source and destination IP Addresses in traffic flows against the database using the IPrep_IS_Malicious and IPrep_threat_category expression operators. These operator can identify traffic and be used to trigger policy feature actions to drop or filter the unwanted to traffic.

Appendix A: Transition from the CNS-318 to CNS-319

Overview:

These steps allow students completing the Part 1 content (CNS-318, Mon-Wed) to transition to the starting state required for Part 2(CNS-319, Thu-Fri).

- **IMPORTANT**: Only run these steps if going from (CNS-318 to CNS-319).
- If starting in the Part 2 (CNS-319) images, skip this procedure.

Estimated time to complete this task: 5 minutes

Procedure to Transition to Start State (CNS-319):

Step	Action
1.	Connect to NS_VPX_01 using the NSIP Address (192.168.10.101) using the PuTTY SSH client. • Use the PuTTY shortcut on the desktop of your Student Desktop OR run the following command: Right Click on **Start > Run > Putty 192.168.10.101** Log into the utility using the following credentials: User Name: **nsroot** Password: **nsroot**

2.	Run the following commands to set the config for the new start state:
	Restore the dependent files for the configuration from part 1(signatures, imports, and SSL certs):
	`batch -filename /var/labstuff/restore/restorefiles_part1end.bat`
	Restore the dependent files for the configuration (signatures, imports, and SSL certs):
	`batch -filename /var/labstuff/restore/restorefiles_part2start.bat`
	Restore the NetScaler configuration:
	`batch -filename /var/labstuff/restore/restoreconf_part2start.bat`
	Reboot the NetScaler:
	`reboot`
	NOTE: This configuration keeps all of the load balancing virtual servers and policies from part 1, with the exception of AppFlow integration with Insight has been removed. The AppQoE and IP Reputation features are disabled and their policies are no longer bound to the associated virtual servers. AppFw feature is disabled, but the policies are still bound to the WebGoat and AFWeb virtual servers, for later demonstrations.
	The transition scripts add an SSL certkey pointing to an expired certificate for *.training.lab. The SSL certkey is in use by two additional lb vservers for WebGoat and AFWeb on SSL and 443.

3.	Reconnect to NS_VPX_01 at 192.168.10.101 using PUTTY. Log on as **nsroot / nsroot**.

4.	Verify the configuration with the following commands:
	`show lb vserver -summary`
	Alternate command:
	`show lb vserver -summary -fullvalues`
	
	Verify all the following load balancing virtual servers are present.
	• lb_vsrv_rbg
	• lb_vsrv_afweb
	• lb_vsrv_webgoat
	• lb_vsrv_callout (will be listed as down)
	• lb_vsrv_afweb_ssl (NEW)
	• lb_vsrv_webgoat_ssl (NEW)

5.	Verify the configuration with the following commands: `show appfw signatures` Verify custom signature "webgoatsigs" appears in list.
6.	Verify the configuration with the following commands: `show lb vserver lb_vsrv_afweb` Verify the appfw policy appfw_pol_afweb is bound to lb_vsrv_afweb.
7.	Verify the configuration with the following commands: `show lb vserver lb_vsrv_webgoat` Verify the appfw policy appfw_pol_webgoat is bound to lb_vsrv_webgoat. Note: If dependencies referenced in the configuration such as Signatures or imported pages are not present on the NetScaler, the depdendent objects such as policy actions or profiles will be missing. Repeat step 2 and run all three scripts to fix the issue and reboot.
8.	Verify the configuration with the following commands: `show responder policy -summary -fullvalues` Verify the three custom responder policies are included in the summary list: • rs_pol_drop_bycallout • rs_pol_drop_bycallout2 • rs_pol_respondwith_err_reqrate
9.	Verify the configuration with the following commands: `show responder action -summary -fullvalues` Verify the one custom responder action is included in the summary list: • rs_act_respondwith_err_reqrate
10.	Save the NetScaler configuration.

11.	Open XenCenter and connect to your assigned XenServer: • Use XenCenter shortcut on Desktop. Shutdown NetScaler Insight Center VM and start NetScaler MAS Virtual Appliance: • Right-click **NS_InsightCenter** in left pane and click **Shutdown**. • Right-click **MAS Virtual Appliance** in left pane and click **Start**, if not running.
12.	Close XenCenter when Virtual Machine operations are complete.

Appendix: Challenge Question Answers

Overview:

This section presents a deeper examination and explanation of some of the decisions made when configuring the Application Firewall Profiles and Start URLs in Exercise 4.1.

Questions & Answers:

Question 1:

Why were AFWeb's and WebGoat's base URLs initially blocked by the default URLs in the basic application firewall profile?

For reference, the default Start URLs are included here:

- Rule 1: ^[^?]+[.](html?|shtml|js|gif|jpg|jpeg|png|swf|pif|pdf|css|csv)$
- Rule 2: ^[^?]+[.](cgi|aspx?|jsp|php|pl)([?].*)?$

And the base URLs referred to are:

- AFWeb (1): http://afweb.training.lab/
- AFWeb (2): http://172.21.10.111/
- WebGoat: http://webgoat.training.lab/WebGoat/attack

Answer 1:

For AFWeb: The base URL for AFWeb terminates in "/" and does not conform to the extensions expected in Rule 1 (basic web content) or the script extensions with or without query strings in Rule 2 (script/query content). Attempts to connect to http://afweb.training.lab/index.htm would succeed with the default URLs.

For WebGoat: The base URL for WebGoat terminates without an extension at all, using /WebGoat/attack. Otherwise, basically the same issue as with AFWeb in that the URL structure does not conform to expected web content or script formats.

Question 2

When updating the AFWeb profile to allow the base URL path ("/") to be allowed, why was the following URL used:

- ^http://afweb[.]training[.]lab/$

As opposed to, this one:

- ^http://afweb[.]training[.]lab/

At this point, the profile is still in "basic" mode and relying on the default Start URLs (without URL Closure).

Answer 2:

This was the narrowest URL that would allow successful navigation to the default path ("/") without automatically granting access to all other content. This ensured unexpected content, like the /allow.demo page or even the .ico files would not be permitted without additional Start URLs to permit access. Content not allowed, would continue to be denied, even if deny URLs hadn't been created yet.

Question 3

For WebGoat: The base URL for WebGoat was added to the Start URL list, and while this allowed navigation to the default page to succeed, all other navigational links failed.

- Why was this addition of the base URL not enough? Especially, if WebGoat navigates using query string parameters.

At this point in the configuration, the following Start URLs are in effect:

- Rule 1: ^[^?]+[.](html?|shtml|js|gif|jpg|jpeg|png|swf|pif|pdf|css|csv)$
- Rule 2: ^[^?]+[.](cgi|aspx?|jsp|php|pl)([?].*)?$
- Rule 3: ^http://webgoat[.]training[.]lab/WebGoat/attack$

Example URLS for WebGoat navigation include:

- http://webgoat.training.lab/WebGoat/attack?Screen=XXXXX&Menu=YYYY
- http://webgoat.training.lab/WebGoat/attack?Screen=XXXXX&Menu=YYYY&otherparams=othervalues

Answer 3:

Even though the WebGoat URL contains query strings, because the URL does not include a standard script extension for cgi/asp/php or other, the URL query strings used for navigation do not confirm to the script/query string content defined in Rule 2. While the custom Rule 3 allowed initial access to WebGoat, all other navigation links failed.

Rule 3 was then replaced with a pattern that allowed content to /WebGoat/attack and borrowed the query portion of the pattern from Rule 2. The updated regular expression deployed for lab purposes allowed the inclusion of any parameter string name-value pair, and not just the use of Screen and Menu pairs. If additional attack content hadn't need to be supported, an alternate URL to limit the types of query parameters allowed might have been used instead.

- Rule 3 (modified): ^http://webgoat[.]training[.]lab/WebGoat/attack([?].*)?$