19

Protecting Against Attacks

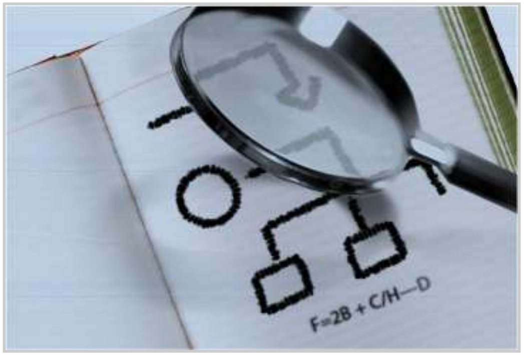
Objectives

After completing this lesson, you should be able to do the following:

- Describe the process of configuring Secure Sockets Layer (SSL)
- Use the `keytool` utility to configure keys and obtain digital certificates
- Configure SSL for the WLS server
- Configure countermeasures for some Web-based attacks such as:
  - Man in the middle
  - Denial of service
  - Large buffer
  - Connection starvation

Road Map

- Protecting the transport layer
  - Secure Sockets Layer (SSL)
  - `keytool`
  - Certificates
  - Configuring SSL
- Protecting against attacks



What Is SSL?

Secure Sockets Layer (SSL) is a protocol that enables:

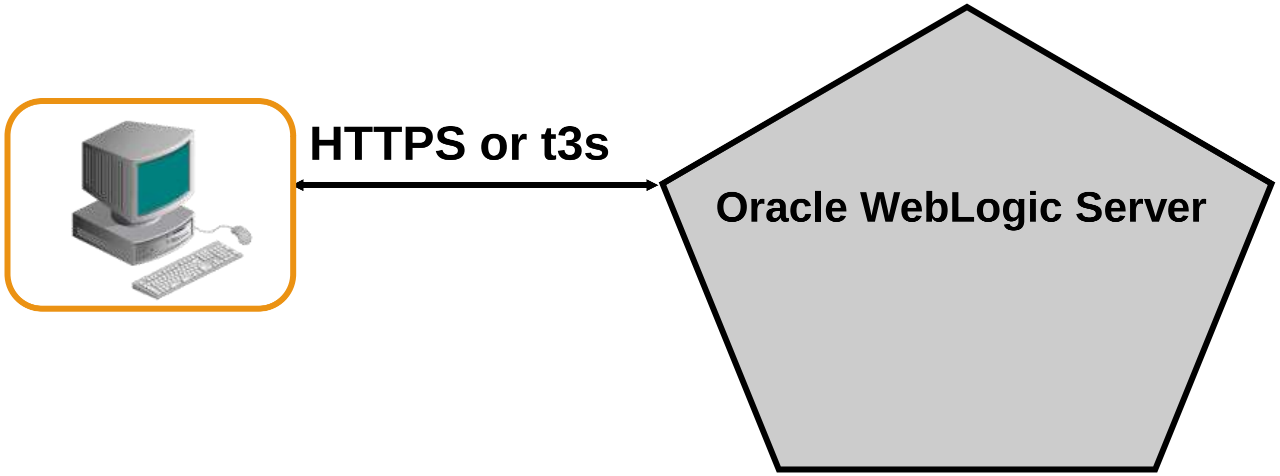
- Connection security through encryption
- A server to authenticate to a client
- A client to authenticate to a server (optional)
- Data integrity such that the data that flows between a client and server is protected from tampering by a third party

Trust and Identity

- SSL and keystore are configured independently.
- For the purpose of backward compatibility, this release of Oracle WebLogic Server supports private keys and a trusted WebLogic Keystore provider.
- Identity:
  - Private key and digital certificate (can now be looked up directly from the keystore, not necessarily as a stand-alone file outside the keystore)
- Trust:
  - Certificates of trusted certificate authorities

Using an SSL Connection

- WLS uses SSL to secure HTTP and t3 communication.
- To use SSL, clients access WLS via the HTTPS or t3s protocols.
  - `https://localhost:7002/orderStock`
  - `t3s://localhost:7002/useCreditCard`



Enabling Secure Communication

- With SSL, data is encrypted using a negotiated symmetric session key.
- A public key algorithm is used to negotiate the symmetric session key.
- In SSL, digital certificates are used to provide a trusted public key.

Oracle WebLogic Server SSL Requirements

To enable Oracle WebLogic Server SSL, you must perform the following steps:

1. Obtain an appropriate digital certificate.
2. Install the certificate.
3. Configure SSL properties.
4. Configure two-way authentication (if desired).
   – SSL impacts performance.

`keytool` Utility

- `keytool` is a standard J2SE SDK utility for managing:
  - The generation of private keys and the corresponding digital certificates
  - Keystores (databases) of private keys and the associated certificates
- The `keytool` utility can display certificate and keystore contents.
- Specify an algorithm different from DSA when generating digital keys using `keytool`.

Obtaining a Digital Certificate: `keytool` Examples

Generate a new self-signed digital certificate:

```
keytool -genkey -alias dwkey -keyalg RSA -keysize 512
    -keystore dw_identity.jks
```

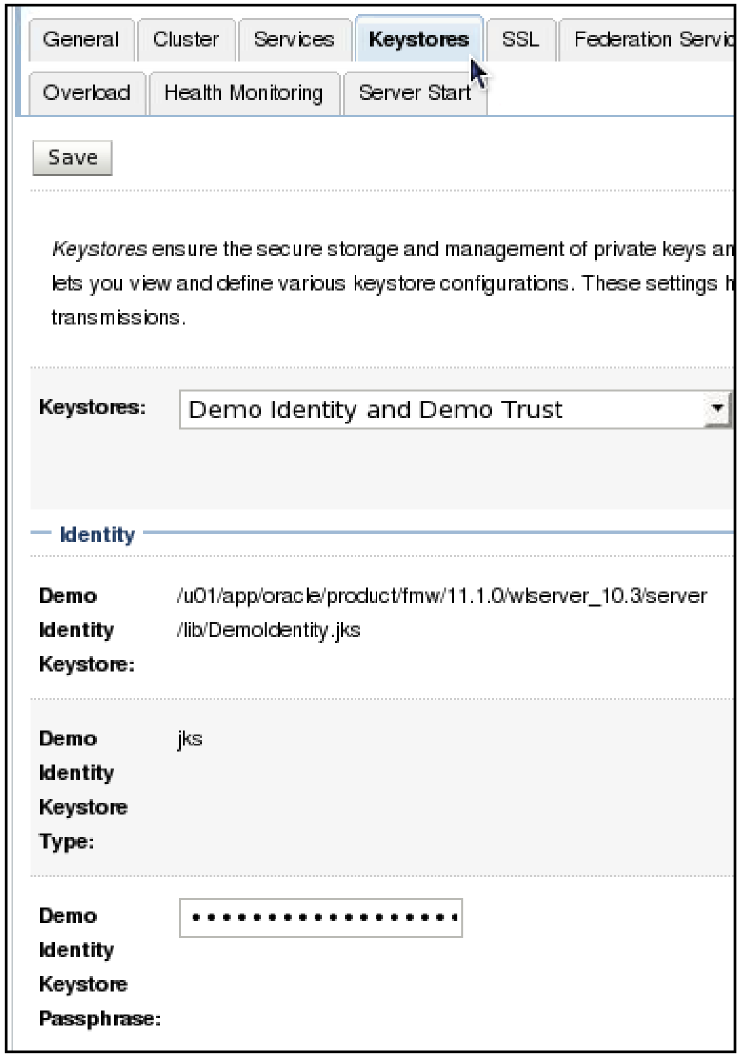
Generate a Certificate Signing Request:

```
keytool -certreq -v -alias dwkey -file
    dw_cert_request.pem
    -keypass dwkeypass -keystore dw_identity.jks
    -storepass dwstorepass
```

Import a signed certificate reply from a CA:

```
keytool -import -alias dwkey -file dw_cert_reply.pem
    -keypass dwkeypass -keystore dw_identity.jks
    -storepass dwstorepass
```

Configuring Keystores



General | Cluster | Services | Keystores | SSL | Federation Servic
Overload | Health Monitoring | Server Start

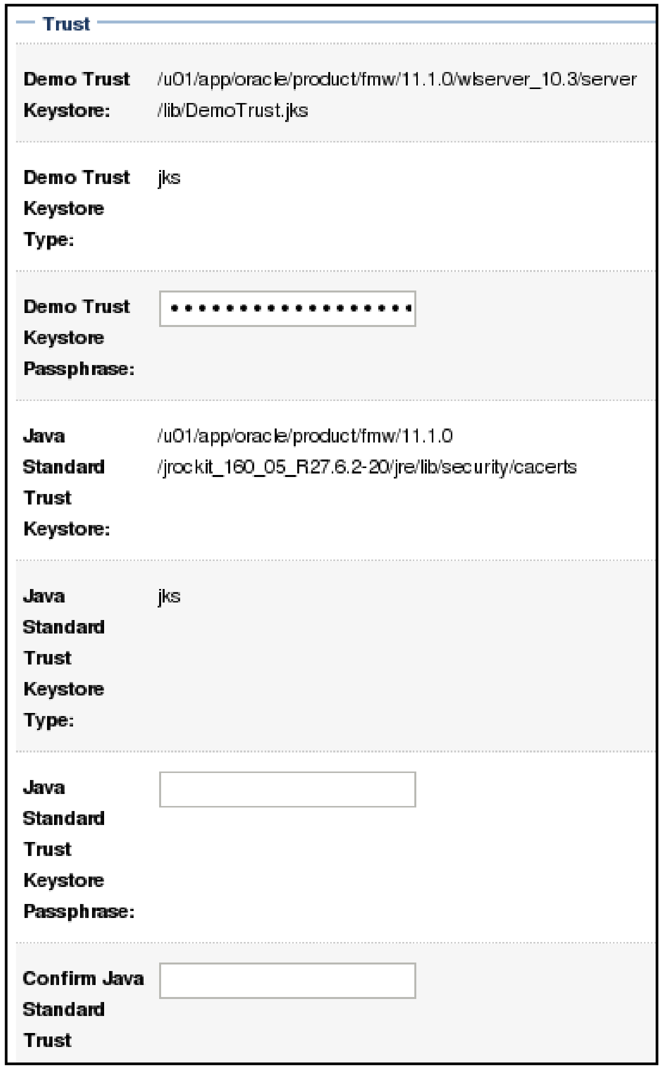
Save

Keystores ensure the secure storage and management of private keys an
lets you view and define various keystore configurations. These settings h
transmissions.

Keystores:	Demo Identity and Demo Trust

Identity

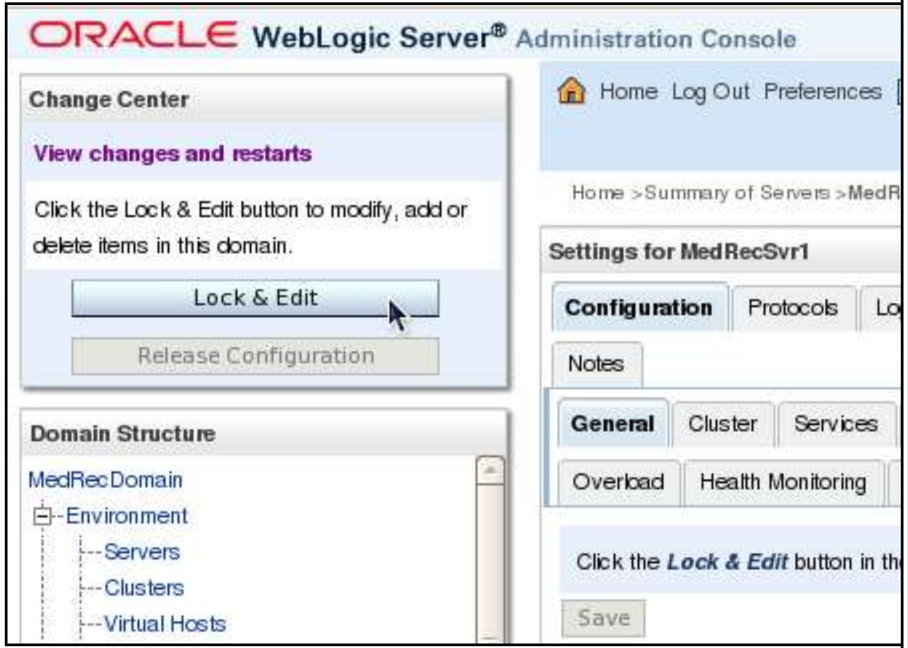
Demo Identity Keystore:	/u01/app/oracle/product/fmw/11.1.0/wlserver_10.3/server /lib/DemoIdentity.jks
Demo Identity Keystore Type:	jks
Demo Identity Keystore Passphrase:	••••••••••••••••••



Trust

Demo Trust Keystore:	/u01/app/oracle/product/fmw/11.1.0/wlserver_10.3/server /lib/DemoTrust.jks
Demo Trust Keystore Type:	jks
Demo Trust Keystore Passphrase:	••••••••••••••••••
Java Standard Trust Keystore:	/u01/app/oracle/product/fmw/11.1.0 /jrockit_160_05_R27.6.2-20/jre/lib/security/cacerts
Java Standard Trust Keystore Type:	jks
Java Standard Trust Keystore Passphrase:	
Confirm Java Standard Trust	

Configuring SSL for an Oracle WebLogic Server



ORACLE WebLogic Server® Administration Console

Change Center

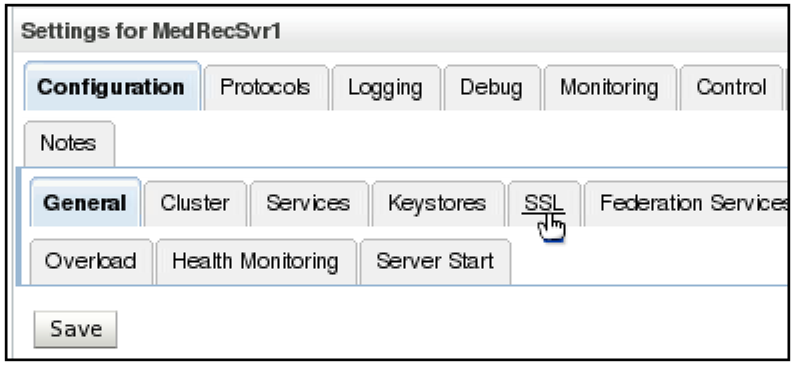
View changes and restarts

Click the Lock & Edit button to modify, add or delete items in this domain.

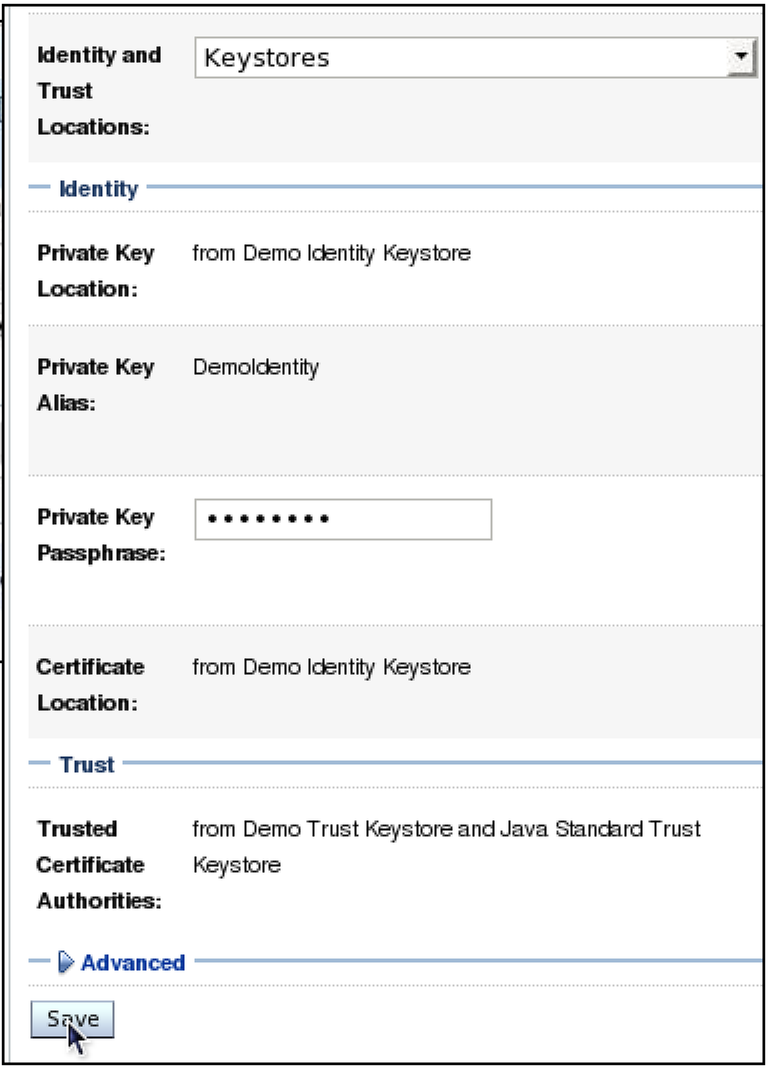
Domain Structure

Settings for MedRecSvr1

Click the *Lock & Edit* button in th



Settings for MedRecSvr1



Identity and Trust Locations:	Keystores

Identity

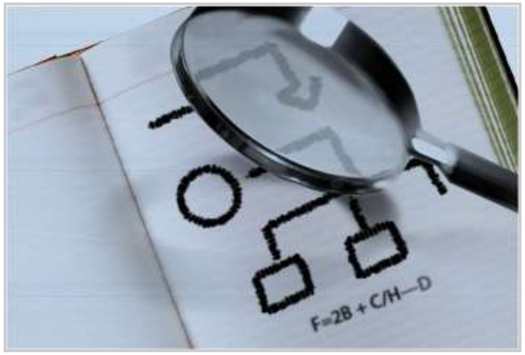
Private Key Location:	from Demo Identity Keystore
Private Key Alias:	DemoIdentity
Private Key Passphrase:	•••••••••
Certificate Location:	from Demo Identity Keystore

Trust

Trusted Certificate Authorities:	from Demo Trust Keystore and Java Standard Trust Keystore

Road Map

- WLS Security Architecture overview
- Users and groups
- Protecting application resources
- Protecting communications
- Protecting against attacks
  - Types of attacks
  - Protecting against man-in-the-middle attacks
  - Protecting against denial of service (DoS) attacks
  - Protecting against large buffer attacks
  - Protecting against connection starvation



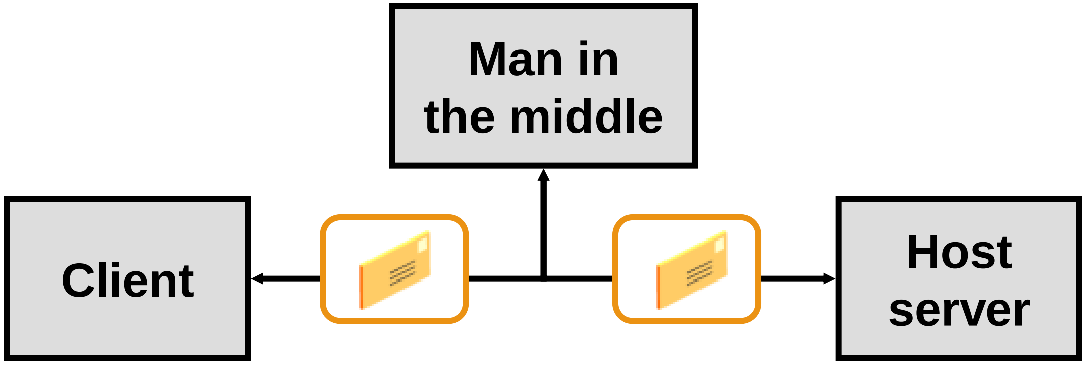
Protecting Against Attacks

WLS can help protect applications against several attacks:

- Man-in-the-middle attacks
- Denial of service (DoS) attacks
- Large buffer attacks
- Connection starvation attacks

Man-in-the-Middle Attacks

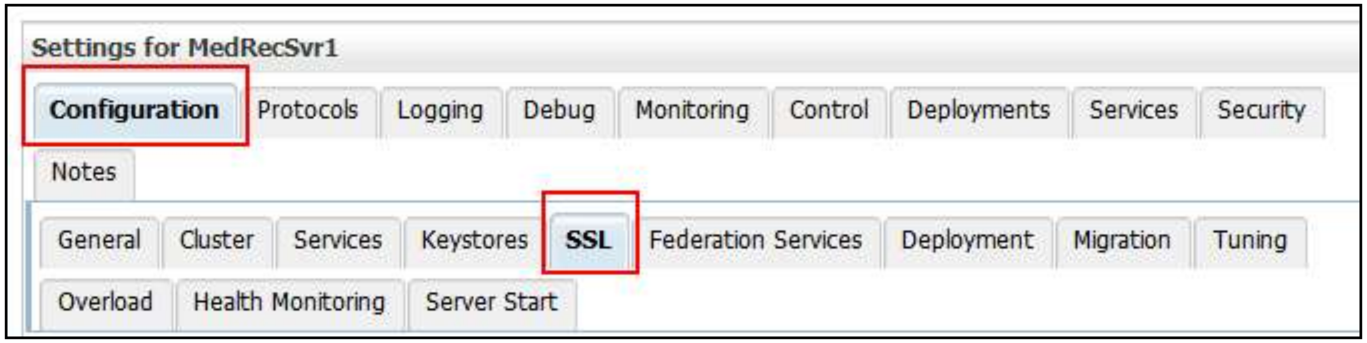
- In the “man-in-the-middle” attack, a third party poses as a destination host intercepting messages between the client and the real host.
- Instead of issuing the real destination host’s SSL certificate, the attacker issues his or her own hoping that the client would accept it as being from the real destination host.

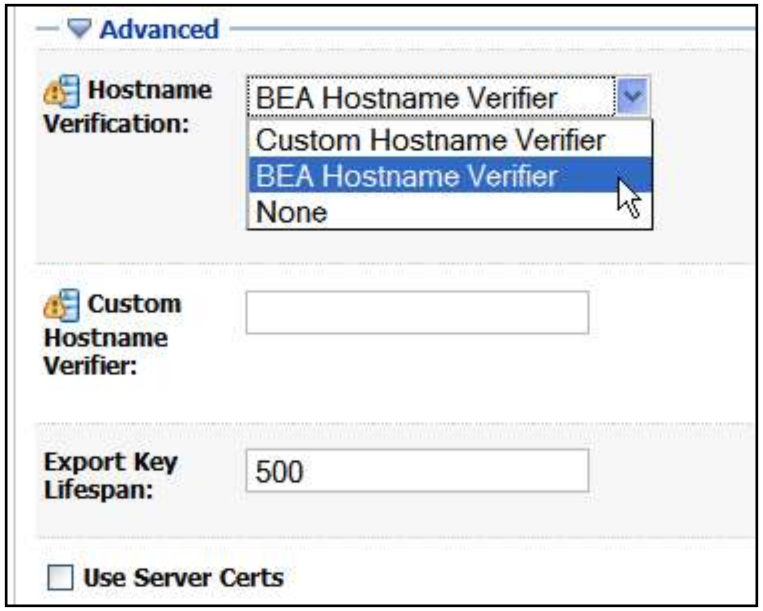


Man-in-the-Middle: Countermeasures

- The “man-in-the-middle” attacks can be resisted by using a Hostname Verifier.
- A Hostname Verifier validates that the host to which an SSL connection is made is the intended or authorized party.
- WLS provides a Hostname Verifier by default.
- A custom Hostname Verifier can be created by implementing the `weblogic.security.SSL.HostnameVerifier` interface.

Configuring a Hostname Verifier


Settings for MedRecSvr1



Hostname Verification: BEA Hostname Verifier

Custom Hostname Verifier:

Export Key Lifespan: 500

Use Server Certs

Denial of Service Attacks

- DoS attacks are attempts by attackers to prevent legitimate users of a service from using that service.
- There are three basic types of attack:
  - Consumption of scarce, limited, or nonrenewable resources
  - Destruction or alteration of configuration information
  - Physical destruction or alteration of network components

Denial of Service Attacks: Countermeasures

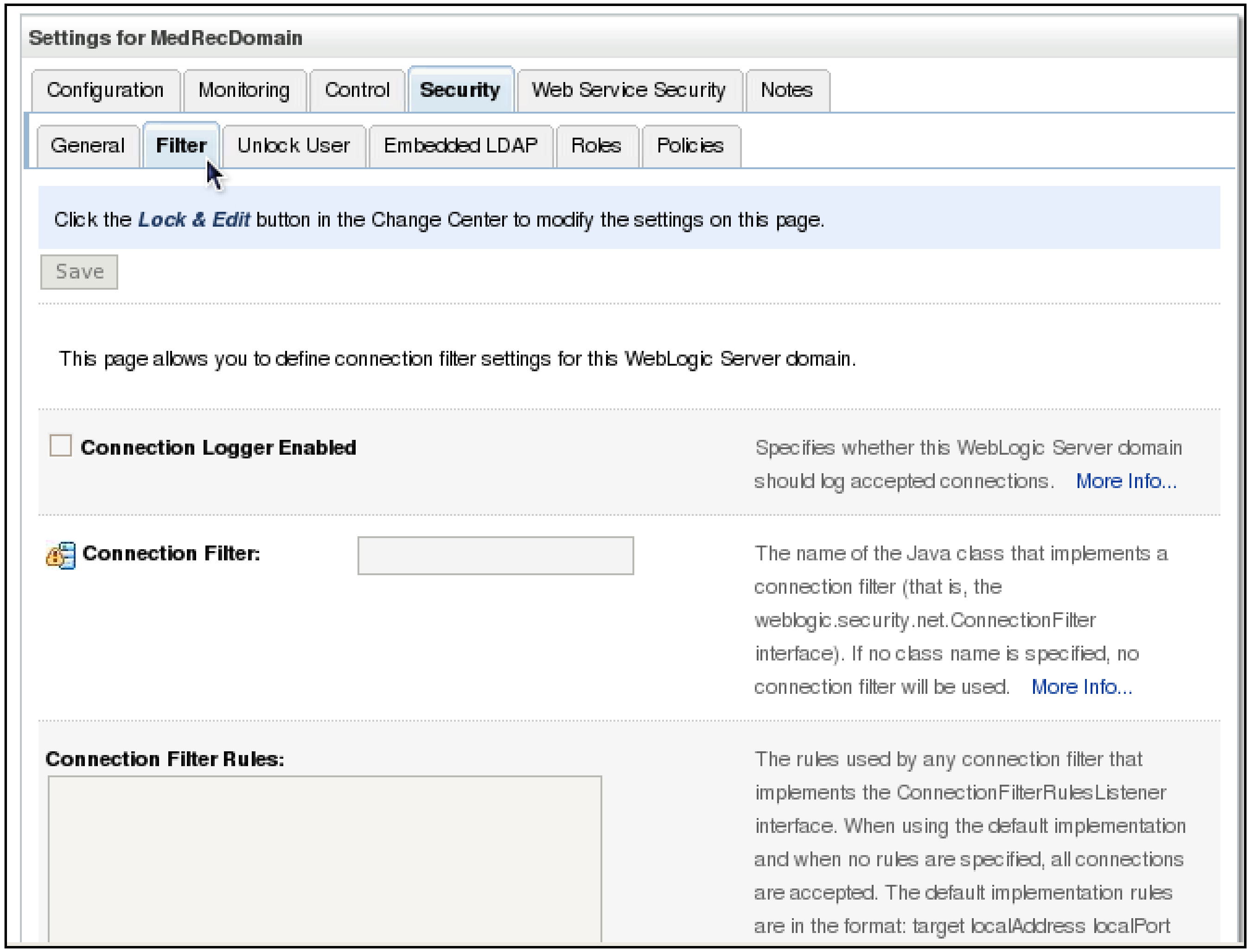
Harden WLS against DoS attacks by:

- Filtering incoming network connections
- Configuring consumable WLS resources with the appropriate threshold and quotas
- Limiting access to configuration information and configuration tools
- Limiting access to back up configuration files
- Preventing unauthorized access by protecting passwords against password-guessing attacks

Filtering Network Connections

- WLS can be configured to accept or deny network connections based on the origin of the client.
- This feature can be used to restrict the:
  - Location from which connections to WLS are made
  - Type of connection made—that is, allow only SSL connections and reject all others
- To filter network connections, create a class that implements the `ConnectionFilter` interface and install it using the Administration Console.

Connection Filter



Settings for MedRecDomain

Click the *Lock & Edit* button in the Change Center to modify the settings on this page.

This page allows you to define connection filter settings for this WebLogic Server domain.

☐ **Connection Logger Enabled** — Specifies whether this WebLogic Server domain should log accepted connections. More Info...

Connection Filter: — The name of the Java class that implements a connection filter (that is, the weblogic.security.net.ConnectionFilter interface). If no class name is specified, no connection filter will be used. More Info...

Connection Filter Rules: — The rules used by any connection filter that implements the ConnectionFilterRulesListener interface. When using the default implementation and when no rules are specified, all connections are accepted. The default implementation rules are in the format: target localAddress localPort

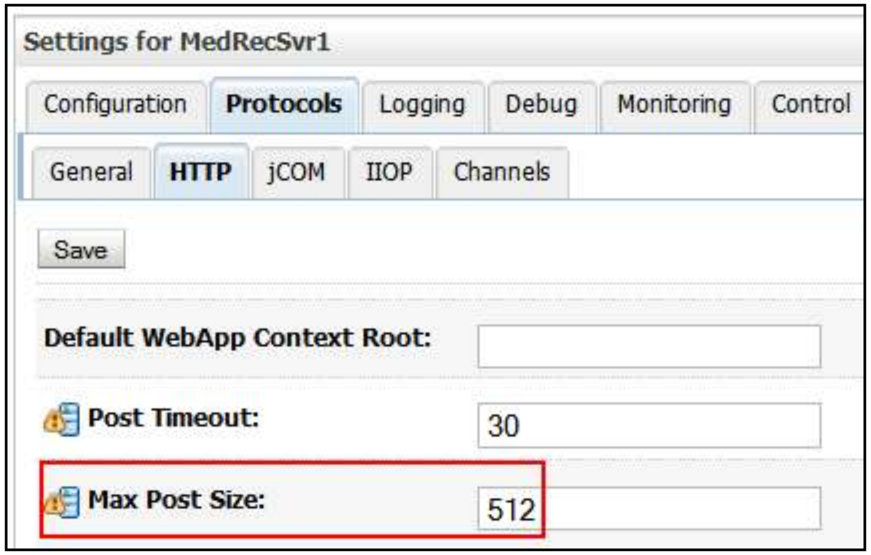
Excessive Resource Consumption

- Denial of service can come from consuming server-side resources used by Web applications:
  - Intentionally generating errors that will be logged, consuming disk space
  - Sending large messages, many messages, or delaying delivery of messages in an effort to cripple JMS
  - Disrupting network connectivity through “connection starvation”
  - Consuming system memory through “large buffer attacks”
- The effect of these attacks can be reduced by setting the appropriate quotas and threshold values.

Large Buffer Attacks

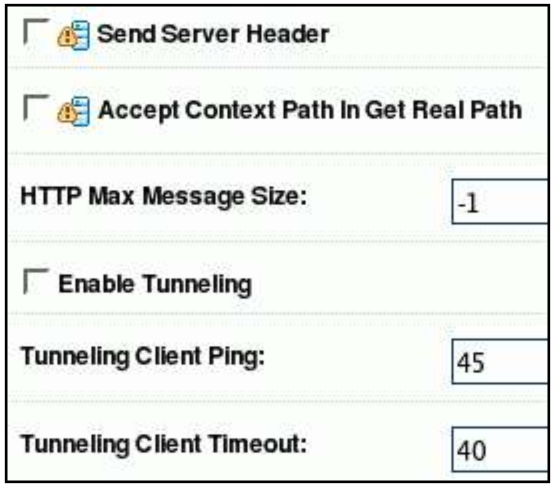
- Individuals can try and bring down a Web site by sending a large buffer of data, which starves the system of memory.
- Administrators can combat this attack by setting a threshold for incoming data.

Setting the Post Size



Settings for MedRecSvr1

Default WebApp Context Root:	
Post Timeout:	30
Max Post Size:	512

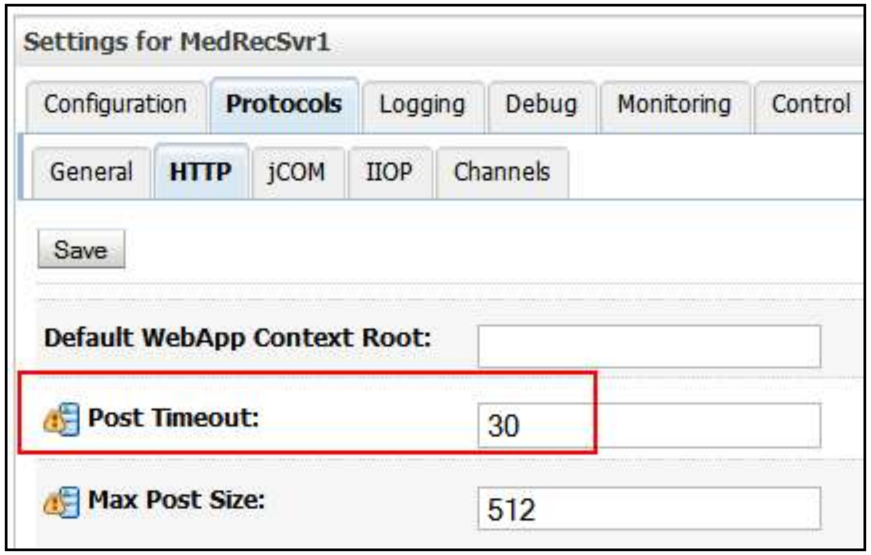


☐ Send Server Header	
☐ Accept Context Path In Get Real Path	
HTTP Max Message Size:	-1
☐ Enable Tunneling	
Tunneling Client Ping:	45
Tunneling Client Timeout:	40

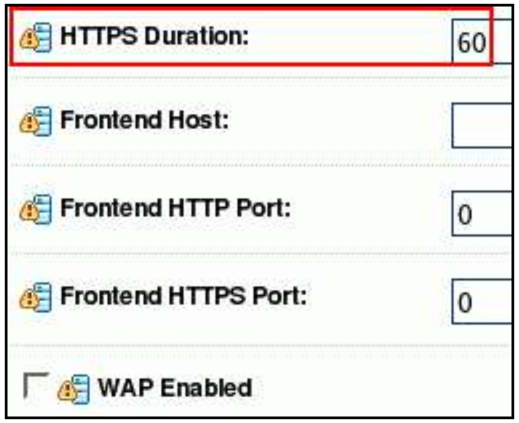
Connection Starvation

- Individuals can try and take down a Web site by sending small, incomplete messages that cause the server to wait.
- Administrators can combat this attack by setting a threshold.
- Connections time out while waiting for the remainder of the data if they have reached the threshold set by the administrator.

Connection Starvation


Settings for MedRecSvr1

Default WebApp Context Root:	
Post Timeout:	30
Max Post Size:	512

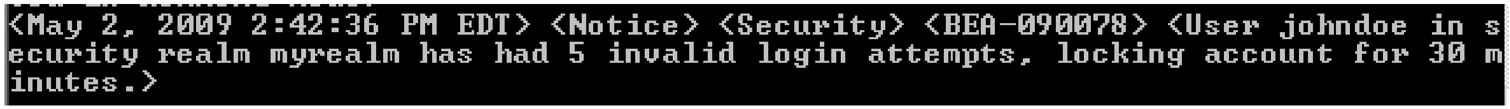


HTTPS Duration:	60
Frontend Host:	
Frontend HTTP Port:	0
Frontend HTTPS Port:	0

☐ WAP Enabled

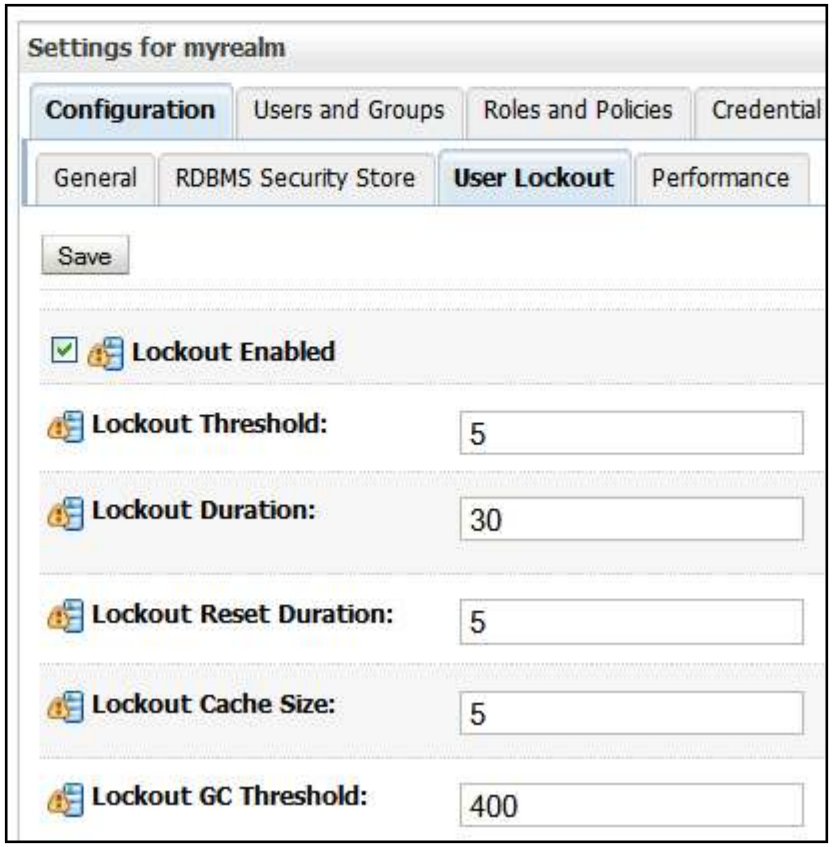
User Lockout

- Individuals attempt to hack into a computer using various combinations of usernames and passwords.
- Administrators can protect against this security attack by setting the lockout attributes.
- The administrator can unlock a locked user using the console.



```
<May 2, 2009 2:42:36 PM EDT> <Notice> <Security> <BEA-090078> <User johndoe in s
ecurity realm myrealm has had 5 invalid login attempts, locking account for 30 m
inutes.>
```

Configuring User Lockout

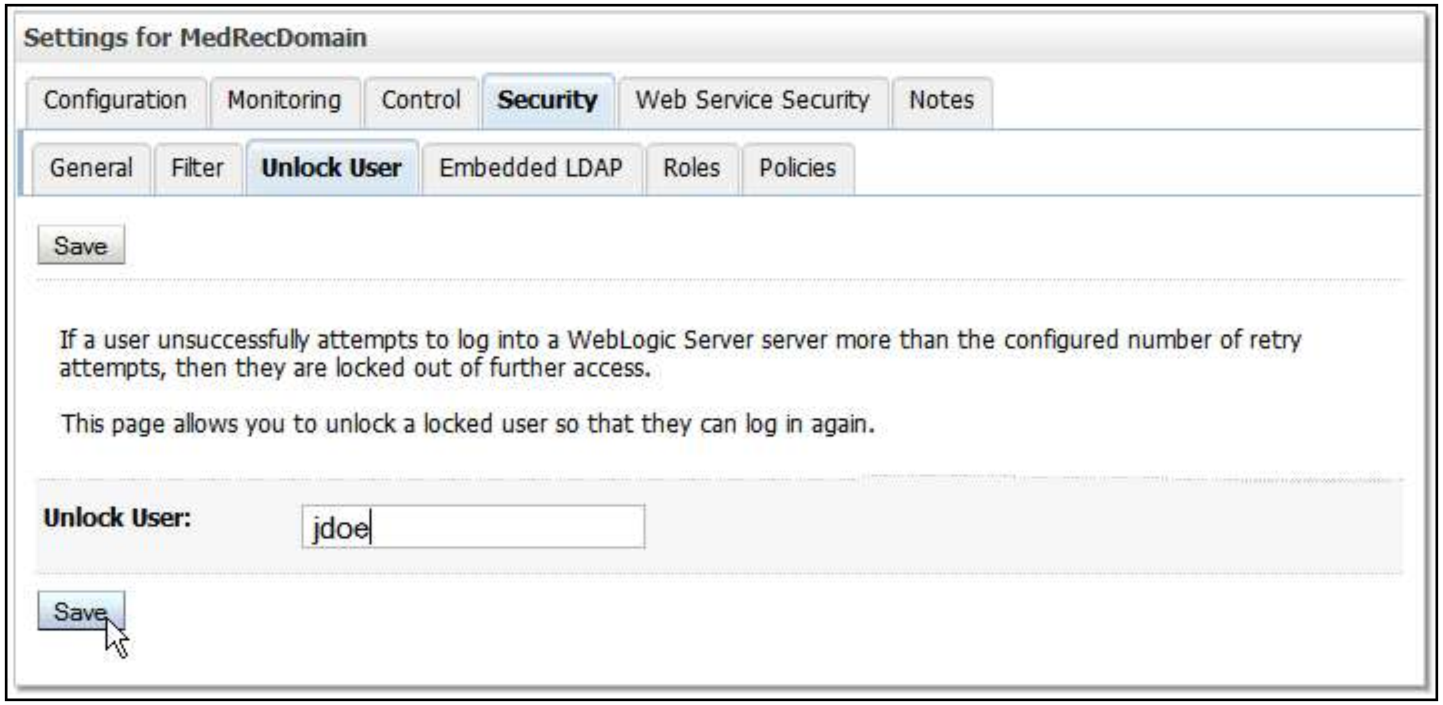


Settings for myrealm

[✓] Lockout Enabled

Lockout Threshold:	5
Lockout Duration:	30
Lockout Reset Duration:	5
Lockout Cache Size:	5
Lockout GC Threshold:	400

Unlocking Users



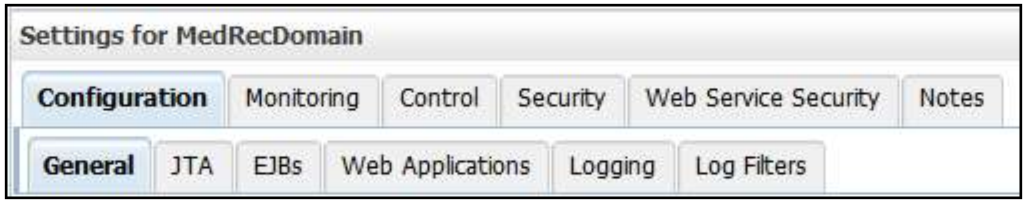
If a user unsuccessfully attempts to log into a WebLogic Server server more than the configured number of retry attempts, then they are locked out of further access.

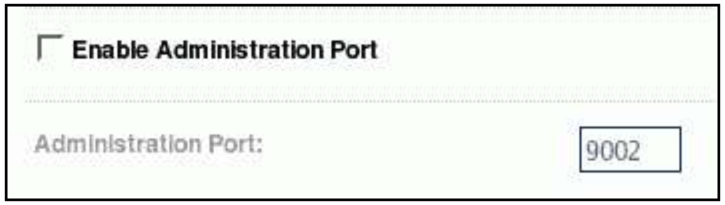
This page allows you to unlock a locked user so that they can log in again.

Unlock User: jdoe

Protecting the Administration Console

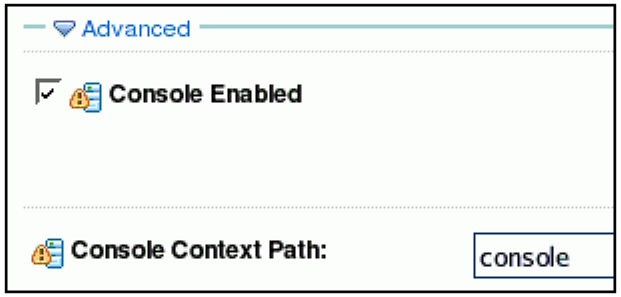
- You can configure a separate administration port for all administration traffic.
- You can change the context path of the console.
- You can disable the console (application).





☐ Enable Administration Port

Administration Port: 9002



☑ Console Enabled

Console Context Path: console

Quiz

The Hostname Verifier is one measure for combating this type of attack:

1. Large buffer
2. Connection starvation
3. Man in the middle
4. User lockout

Quiz

To counter connection starvation attacks, you can set:

1. Max Post Size
2. Post Timeout
3. Hostname Verifier
4. User lockout

Summary

In this lesson, you should have learned how to:

- Describe the process of configuring SSL
- Use the `keytool` utility to configure keys and obtain digital certificates
- Configure SSL for the WLS server
- Configure countermeasures for some Web-based attacks such as:
  - Man in the middle
  - Denial of service
  - Large buffer
  - Connection starvation

ORACLE

Practice 19: Overview

This practice covers the following topics:

- Using `keytool` to generate an identity keystore that contains a private key and a self-signed public certificate
- Configuring keystores using the Administration Console
- Configuring SSL for a managed server