Oracle Weblogic Server Handbook

V 4.0

DigiTalk

https://digitalksystems.com/

Reach us at digitalk.fmw@gmail.com
DigiTalk Channel: https://www.youtube.com/channel/UCCGTnI9vvF_ETMhGUXGdFWw
Playlists: https://www.youtube.com/@digitalk.middleware/playlists
Weblogic Server Architecture: https://youtu.be/gNqeIfLjUqw

Table of Contents

1. About WebLogic Server

WebLogic Server is a popular Java-based application server developed by Oracle. It is used for building and deploying enterprise Java applications, providing a robust and scalable environment for hosting web services, Java EE applications, and other distributed applications. WebLogic Server offers features for high availability, scalability, security, and management, making it suitable for mission-critical applications.

WebLogic Server is part of the Oracle Fusion Middleware family and serves as a middleware platform for deploying, managing, and running Java applications.

It provides various services, including:

- **Servlet and JSP Container:** It hosts and manages Java web applications, including Servlets and Java Server Pages (JSP).
- **EJB Container:** It supports Enterprise JavaBeans (EJB) components for building enterprise-level, distributed applications.
- **JMS (Java Messaging Service):** It enables messaging between various components of your application and external systems.
- **Connection Pooling:** WebLogic Server manages a pool of database connections, improving efficiency and scalability.
- **Security:** It offers robust security features, including authentication, authorization, and encryption.
- **Clustering:** You can create clusters of WebLogic Servers to provide high availability and load balancing for your applications.
- **Deployment:** WebLogic allows you to deploy applications in various formats, such as WAR, EAR, and JAR files.
- **Management and Monitoring:** It offers management tools and a web-based console for configuring, monitoring, and managing server instances.

WebLogic Server provides many configuration options and advanced features, making it suitable for a wide range of enterprise applications, from simple web apps to complex, distributed systems. It is typically used in large-scale enterprises to ensure the reliability and performance of Java-based applications.

1.1.2 Weblogic Server Editions: Standard, Enterprise and Suite

WebLogic Server comes in several different editions, each tailored to different needs and requirements. Here's a comparison of WebLogic Server Standard Edition, WebLogic Server Enterprise Edition, and WebLogic Server Suite:

WebLogic Server Standard Edition:

- **Features:** WebLogic Server Standard Edition provides core features for building and deploying Java EE applications. It includes support for Java EE APIs and standards like Servlets, JSP, EJB, JMS, JTA, and more.
- **Use Cases:** It is suitable for small to medium-sized applications and projects that require Java EE compliance and basic application server capabilities.
- **Scalability:** It is limited in terms of scalability and clustering capabilities compared to the Enterprise Edition.
- **Management Tools:** Standard Edition provides basic management and monitoring tools but may lack some advanced features found in higher editions.
- **Licensing:** Licensing costs are generally lower compared to the Enterprise and Suite editions.

WebLogic Server Enterprise Edition:

- **Features:** WebLogic Server Enterprise Edition includes all the features of the Standard Edition and adds more advanced features for building and managing large-scale, mission-critical applications.
- **Clustering and High Availability:** It offers robust clustering, load balancing, and failover capabilities, making it suitable for high-traffic and high-availability scenarios.
- **Advanced Management and Monitoring:** Enterprise Edition provides advanced management and monitoring tools, making it easier to configure, monitor, and troubleshoot server instances and applications.
- **Security:** It includes advanced security features such as single sign-on (SSO), role-based access control, and encryption.

- **Use Cases:** Enterprise Edition is a suitable choice for larger and more complex enterprise applications that require high availability, scalability, and advanced management features.
- **Licensing:** It typically has a higher licensing cost compared to the Standard Edition.

WebLogic Server Suite:

Features: WebLogic Server Suite includes all the features of the Enterprise Edition and adds additional components and services such as Oracle Coherence (in-memory data grid), Oracle WebLogic Server Diagnostic Framework, and Oracle Traffic Director (a software load balancer).

- **Use Cases:** It is designed for building and managing comprehensive and high-performance enterprise solutions. It is often chosen for large-scale, mission-critical applications and complex infrastructures.
- **Scalability and Performance:** The Suite edition provides advanced features for improving application scalability, performance, and fault tolerance.
- **Components:** It includes additional components that can enhance application performance, scalability, and fault tolerance.
- **Licensing:** WebLogic Server Suite is the most expensive of the three editions due to its inclusion of additional components and services.

In summary, WebLogic Server Standard Edition is suitable for smaller applications and projects, while WebLogic Server Enterprise Edition is designed for larger and more complex enterprise applications with advanced requirements. WebLogic Server Suite includes everything in the Enterprise Edition and adds more components and services for building comprehensive and high-performance enterprise solutions. The choice of edition depends on the specific needs, scale, and budget of the project or organization.

2.1.2 Reverse Proxy

A reverse proxy is a server or software component that sits between client devices and web servers. It acts as an intermediary that receives client requests and forwards them to the appropriate backend server. Reverse proxies are used to perform various functions, including load balancing, security, caching, and SSL termination, to improve the performance, security, and reliability of web applications. Here's an explanation of the concept with an example:

Concept of a Reverse Proxy:

Load Balancing: One of the primary purposes of a reverse proxy is to distribute incoming client requests across multiple backend servers. This load balancing ensures that no single server becomes overwhelmed with traffic and helps improve the scalability and fault tolerance of a web application.

Security: Reverse proxies can provide an additional layer of security by protecting the backend servers from direct exposure to the internet. They can filter and inspect incoming requests, mitigate DDoS attacks, and enforce security policies.

Caching: Reverse proxies can cache responses from backend servers. Caching frequently accessed content can significantly reduce server load and improve response times for clients. This is especially useful for static assets like images, CSS, and JavaScript files.

SSL Termination: A reverse proxy can handle SSL/TLS encryption and decryption, offloading this processing from the backend servers. This is called SSL termination and can improve server performance and simplify certificate management.

Content Compression: Reverse proxies can compress responses before sending them to clients, reducing bandwidth usage and improving page load times.

Example of a Reverse Proxy:

Let's consider an example of a reverse proxy in a web application deployment:

Scenario: E-commerce Website with a Reverse Proxy

Suppose you are running a popular e-commerce website with multiple web servers serving your online store. You want to ensure high availability, improve security, and enhance performance. Here's how a reverse proxy can help:

Load Balancing: You set up a reverse proxy server (e.g., Nginx or HAProxy) in front of your web servers. The reverse proxy distributes incoming client requests evenly across these servers. If one server becomes overloaded, the reverse proxy can automatically route requests to less busy servers.

SSL Termination: The reverse proxy handles SSL/TLS encryption and decryption. This offloads the CPU-intensive SSL processing from your web servers and simplifies certificate management. Clients connect securely to the reverse proxy, which then communicates with your backend servers over HTTP.

Caching: The reverse proxy caches frequently requested images, CSS, and JavaScript files. When a client requests these assets, the reverse proxy can serve them directly from the cache, reducing the load on your web servers and speeding up page load times.

Security: The reverse proxy acts as a shield, protecting your web servers from direct exposure to the internet. It can inspect and filter incoming requests, block malicious traffic, and help prevent common web attacks like SQL injection or XSS.

By using a reverse proxy in this scenario, you can achieve better scalability, performance, and security for your e-commerce website while ensuring high availability and reliable access for your customers.

2.1.3 Demilitarized Zone (DMZ)

A Demilitarized Zone (DMZ) is a network configuration commonly used in computer security to segregate and isolate a part of a network from the internal, trusted network and the untrusted external network (usually the internet). The primary purpose of a DMZ is to add an extra layer of security by placing systems that need to be publicly accessible but pose a potential security risk, such as web servers or email servers, in an isolated zone.

Concept of DMZ:

Imagine you have a network that includes both internal resources (e.g., employee workstations and databases) and externally accessible services (e.g., a web server). Placing the web server directly on the internal network can pose a security risk. If the web server were to be compromised, an attacker might gain access to your internal network, including sensitive data.

A DMZ is used to create a separate network segment, often called the DMZ network, where publicly accessible services are located. The DMZ is isolated from the internal network and protected by a firewall. This way, if a web server in the DMZ is compromised, the attacker is still isolated from the internal network.

Example of a DMZ:

Let's consider a simple network setup with a DMZ, using a web server as an example:

- **Network Components:**
  - Internal Network: This is the trusted network where your organization's sensitive resources, such as employee workstations and databases, are located.
  - DMZ Network: This is the partially trusted network where externally accessible services reside. In this example, we have a web server.
  - External Network (Internet): This is the untrusted network where external users access your web server.
- **Firewall Configuration:**
  - A firewall separates the internal network from the DMZ network and the DMZ network from the external network.

- Access control rules are configured on the firewall to control traffic between these segments. For example, the firewall allows traffic from the internet to the web server in the DMZ but prevents direct access to the internal network.

❖ **Web Server in the DMZ:**

- The web server hosting your public website is placed in the DMZ network.
- This web server is configured to serve content to external users over the internet.

❖ **Internal Resources:**

- Resources like databases, employee workstations, and sensitive data remain in the internal network.
- Access to these resources is strictly controlled through the firewall, which only permits specific, authorized traffic.

In this example, the DMZ acts as a buffer zone that isolates the publicly accessible web server from the internal network. Even if an attacker were to compromise the web server in the DMZ, they would have a much more challenging time trying to breach the internal network.

The DMZ concept is not limited to web servers; it can be extended to other services like email servers, DNS servers, or any service that needs to be accessible from the internet. The key is to carefully control and monitor the traffic between the DMZ and the internal network to maintain security and isolation.

2.1.4 High Availability, Scalability and Maintainability

High Availability, Scalability, and Maintainability are essential concepts in the design and operation of computer systems, particularly in the context of web services and distributed systems. Here's an explanation of each term with examples:

High Availability:

High Availability (HA) is a measure of a system's ability to remain operational and accessible for an extended period with minimal downtime. HA is crucial for systems that need to provide continuous service, such as web applications or critical infrastructure.

Example: Consider a popular e-commerce website. High Availability ensures that the website remains accessible to customers 24/7, even during heavy traffic or server failures. To achieve this, the system might use load balancing, redundant servers, and failover mechanisms. If one server becomes unavailable, another takes its place, ensuring the website's continuous availability.

Scalability:

Scalability is the system's capability to handle increased workload or resource demands by adding more resources or components. It ensures that a system can grow to accommodate additional users, data, or traffic without a significant drop in performance.

Example: A social media platform experiences a surge in users due to a viral trend. Scalability allows the platform to handle the increased load by adding more servers, storage, and network resources. As user numbers grow, the system can scale horizontally (adding more servers) or vertically (upgrading existing servers) to maintain responsiveness.

Maintainability:

Maintainability refers to a system's ease of maintenance, including making updates, fixing bugs, and managing hardware or software changes. A highly maintainable system is cost-effective to manage and can quickly adapt to evolving requirements.

Example: An enterprise software application used by a large organization. Maintainability ensures that updates and patches can be applied without causing downtime or disrupting daily operations. It also means that the system's code is well-documented and modular, making it easier for developers to understand, modify, and fix issues.

In many real-world systems, achieving a balance among these three factors is essential. For instance, a highly available system may require scalability and maintainability to ensure it can adapt to changing conditions and remain reliable. Similarly, a highly scalable system should also be maintainable to handle continuous growth without increasing operational complexity.

Here's an example that combines these concepts:

Content Delivery Network (CDN):

CDNs are designed for high availability, scalability, and maintainability:

High Availability: A CDN ensures that content (e.g., images, videos) is readily available and delivered from the closest server to the user. If one server or data center fails, other servers take over, maintaining content accessibility.

Scalability: CDNs can distribute content globally to millions of users. As user demand grows, more servers and data centers can be added to accommodate the increased traffic and maintain fast delivery.

Maintainability: CDNs are designed for easy content updates and changes. Content can be updated at the origin server, and the changes propagate across the CDN quickly. Additionally, CDN providers maintain server infrastructure, reducing maintenance burden for clients.

In this example, a CDN combines high availability, scalability, and maintainability to deliver content efficiently and reliably to users worldwide.

2.1.9. Why we use Web Server when we already have Hardware Load Balancer

Using a web server in front of a WebLogic Server, even when an external load balancer is in place, can provide several benefits and is a common architectural choice for many enterprise applications.

Here are some reasons why you might use both a web server and an external load balancer in your setup:

Request Handling and Routing:

A web server can efficiently handle static content and serve as a reverse proxy for requests, offloading this work from WebLogic Server. This allows the load balancer to focus on routing application-specific requests, while the web server handles requests for images, stylesheets, and other static assets.

Security and SSL Termination:

Web servers are often equipped with strong security features, including SSL termination. This means the web server can handle SSL/TLS encryption and decryption, reducing the SSL/TLS processing overhead on WebLogic Server. This can enhance performance and simplify SSL certificate management.

Content Caching:

Web servers can cache static content and serve cached content to clients, reducing the load on both the web server and WebLogic Server. This improves response times for clients and reduces the load on the application server.

HTTP Acceleration and Compression:

Web servers can accelerate HTTP responses by applying compression to reduce the size of web content. This can lead to faster content delivery and improved performance for clients.

Content Rewriting and Transformation:

Web servers can rewrite URLs and transform content to adapt it for different client devices, making it easier to serve web content to a variety of browsers and devices.

Load Balancer Health Checks:

External load balancers can perform health checks at the application level to ensure that WebLogic Server instances are responsive and healthy. However, web servers can also perform basic health checks on behalf of the load balancer to ensure that the application server is up and running. This provides an additional layer of redundancy and failover capability.

Easier Integration with Legacy Systems:

In some cases, legacy systems or older applications may require a web server to handle certain protocols or authentication methods, which can be seamlessly integrated with modern web applications running on WebLogic Server.

Security and Firewall Considerations:

Placing a web server in front of WebLogic Server can also help in firewall configuration and access control. You can configure the firewall to allow traffic only to the web server, and the web server can then route requests to the application server within the network, enhancing security.

In summary, using a web server in conjunction with an external load balancer provides a layered approach to improving the performance, security, and scalability of your web applications. While the external load balancer primarily handles load distribution, the web server offers a wide range of optimizations for web content, security, and content management. Together, they create a more robust and efficient architecture for delivering web applications to end users.

5.1.2 IMPACT IN CASE OF ADMIN SERVER IS DOWN

The Oracle WebLogic Server Administration Server (admin server) is a critical component in the management and administration of a WebLogic domain. If the admin server goes down or becomes unavailable, it can have significant impacts on the operation of the WebLogic environment. Here are some of the key impacts when the admin server is down:

Limited Configuration Changes:
You won't be able to make configuration changes to the domain. This includes adding or removing managed servers, configuring data sources, deploying applications, and modifying security settings. Configuration changes are typically made through the WebLogic Administration Console, which relies on the admin server.

Deployment and Undeployment Issues:
You cannot deploy or undeploy applications or resources in the domain. This can hinder the ability to update applications or libraries and add new resources to the environment.

No User and Role Management:
User authentication and authorization may not function correctly. New users and roles cannot be added or modified. This can impact security and access control within the domain.

Server Control Limitations:
You won't be able to start, stop, or restart managed servers or clusters through the WebLogic Administration Console. This can affect server availability and maintenance.

Logging and Log Management Issues:
Log files generated by managed servers will continue to be produced, but you may have limited access to them through the Administration Console. Centralized log management is typically done through the admin server.

Monitoring and Diagnostics Limitations:
Real-time monitoring and diagnostics tools provided by the WebLogic Administration Console will be unavailable. You won't have a centralized view of server metrics, performance data, and diagnostic information.

Security Configuration and Management Challenges:
Making changes to security settings, including configuring authentication providers, authorization policies, and SSL settings, may not be possible. This can have implications for the security of the environment.

Backup and Recovery Difficulties:
You may face challenges in managing backups and disaster recovery. The admin server is responsible for creating and maintaining backups of the domain configuration and state.

Change Management and Version Control Issues:
Tracking configuration changes and managing version control may not be possible. The admin server typically provides tools for detecting changes, comparing configurations, and rolling back to previous states.

Console and GUI Unavailability:
The WebLogic Administration Console, which provides a user-friendly interface for administrators, will not be accessible. Administrative tasks will need to be carried out using command-line tools or scripts.

Limited Scripting and Automation:
Automation of administrative tasks using scripting tools like WebLogic Scripting Tool (WLST) may not work as expected since these scripts are typically executed through the admin server.

Clustering and Load Balancing Challenges:
Managing server clusters and ensuring proper load balancing may be impacted, as these configurations are typically managed through the admin server.
To mitigate these impacts, it's important to monitor the health of the admin server, implement high availability measures (e.g., configuring multiple admin servers), and establish a disaster recovery plan that includes procedures for restoring the admin server in case of a failure. Ensuring the stability and availability of the admin server is crucial for maintaining a well-functioning and secure WebLogic environment.

8.1.3. USE CASES OF STAGE, NO-STAGE AND EXTERNAL-STAGE

WebLogic Server offers three deployment modes—Stage, No-Stage, and External Stage—to cater to different use cases and deployment scenarios. Each mode has its own advantages and is suitable for specific situations. Here are explanations and use cases for each deployment mode:

Stage Mode Deployment:

Use Cases:

- Standard Application Deployment: Stage mode is commonly used for standard application deployment scenarios. In this mode, the application or resource is copied to the server's staging directory before deployment.
- Development and Testing Environments: Stage mode is ideal for development and testing environments, as it provides a straightforward and predictable deployment process.
- Situations with Sensitive Application Updates: If you need to ensure that the deployed application is not directly modified by external processes, you can use Stage mode.

Advantages:

- The original application archive remains unchanged, reducing the risk of accidental modifications.
- Deployment is more controlled, as the server works with a copy of the application, which minimizes potential issues during deployment.
- It allows for easy version management, as different versions of the same application can coexist in the staging directory.

No-Stage Mode Deployment:

Use Cases:

- Production Environments: No-Stage mode is often preferred for production environments, where deployment time is critical, and you want to avoid unnecessary copying of large application archives.
- Frequent Application Updates: When applications need to be updated frequently, using No-Stage mode can save time and reduce storage overhead.

- Applications on Remote or Shared Locations: If your application archives are stored on remote or shared locations, No-Stage mode can deploy them directly without copying.

Advantages:

- Faster deployment process because the application is used directly from its original location, reducing copying time.
- Reduced storage overhead, especially when dealing with large applications that don't need to be copied to the server's local staging directory.
- Well-suited for situations where staging doesn't provide significant benefits.

External Stage Mode Deployment:

Use Cases:

- Complex Deployment Scenarios: External Stage mode is useful when dealing with complex deployment scenarios where an external process or script handles application staging.
- Custom Deployment Workflows: If your organization has custom deployment workflows that involve external tools or processes for staging applications, External Stage mode is appropriate.
- Shared Staging Across Multiple Servers: In situations where multiple WebLogic servers need to deploy the same pre-staged application, External Stage mode ensures consistency.

Advantages:

- Allows for integration with custom deployment scripts or third-party tools for application staging.
- Useful when applications are pre-staged and synchronized across multiple servers.
- Provides flexibility for organizations with specific deployment processes and needs.

It's important to choose the right deployment mode based on your specific use case, considering factors such as the deployment frequency, application size, deployment speed requirements, and existing deployment workflows. The flexibility offered by these deployment modes allows WebLogic Server to cater to a wide

15.1 PATCHING METHODS

Different Utilities to Apply Patches:

- BSU (BEA Smart Update or Oracle Smart Update)
- OPatch
- SPB (Stack Patch Bundle)

Which utility is applicable for which weblogic server is depend on which version of weblogic you are using:

- Weblogic Server 9.x and higher versions till 12.1.1
- Weblogic Server 12.1.2 and afterwards releases

In a nutshell –

- Weblogic Server 12.1.1 and earlier releases use Smart Update (BSU) to apply patches. So If you are using weblogic 12.1.1 or earlier versions, then after you have obtained the patch, you should apply it using the BSU (Smart Update tool), bsu.cmd for Windows or bsu.sh for UNIX based operating systems)
- Weblogic Server 12.1.2 and all release afterwards use OPatch & SPB to apply patches.

DISCLAIMER AND CONSENT

This document is being provided by DigiTalk as part of its effort to assist users in understanding and working with Oracle WebLogic Server. The Company wishes to emphasize that this document is not affiliated with Oracle Corporation ("Oracle") in any way, and the content contained herein is based solely on publicly available product documentation provided by Oracle.

While every effort has been made to ensure the accuracy and reliability of the information presented in this document, there is a possibility of typographical errors or inaccuracies. DigiTalk does not guarantee the correctness or completeness of the content provided in this document.

Users of this document are encouraged to cross-reference the information presented here with Oracle's official documentation available on their website or other authoritative sources. Any discrepancies or inaccuracies found in this document should be reported to us at digitalk.fmw@gmail.com.

By using this document, you acknowledge and consent to the following:

This document is not officially endorsed or verified by Oracle.

The Company makes no claims or guarantees about the accuracy or suitability of the information contained in this document.

Users are responsible for verifying and validating any information presented here for their specific use case.

DigiTalk disclaims any liability for any errors, omissions, or damages that may result from the use of this document.

If you discover any inaccuracies or errors in this document, please report them to digitalk.fmw@gmail.com, and the Company will endeavor to correct them as necessary.

This consent statement is provided to ensure transparency and understanding of the limitations of the information contained in this document. By using this document, you agree to abide by the terms and conditions outlined herein.