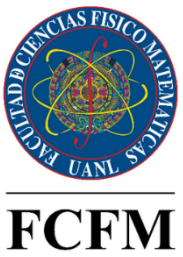

Universidad Autónoma de Nuevo León

Facultad de Ciencias Físico Matemáticos

Sistemas Operativos

“Escenario práctico de migración a Linux”

Jorge Alberto Canales Garduño 1808518

Profesor: Jorge Alberto Islas Pineda

Jueves, 08 de mayo del 2025

Caso práctico: Migración de servidores web empresariales

Escenario: Una empresa mediana está migrando su aplicación web heredada de un antiguo servidor CentOS 6 a un nuevo servidor Rocky Linux 8. La aplicación consta de:

- Apache HTTPD 2.2 (se actualizará a 2.4)
- MySQL 5.1 (a actualizar a MariaDB 10.3)
- Aplicación PHP 5.6 personalizada (compatible con PHP 7.4)
- Cron jobs y shell scripts heredados
- Reglas de cortafuegos personalizadas y políticas SELinux

Fase 1: Planificación Pre-Migración

I. Inventory Assessment

1. Para documentar todos los servicios, dependencias y configuraciones actuales, realizaría un inventario detallado del sistema usando comandos como *systemctl, chkconfig, ps aux, netstat, ss, y rpm -qa* para listar servicios activos, procesos y paquetes instalados. También revisaría archivos de configuración clave ubicados en */etc/* (por ejemplo, *httpd.conf, my.cnf, php.ini, crontab*, y scripts personalizados), así como políticas SELinux (*/etc/selinux/*) y reglas del firewall *(iptables* o *firewalld*). Toda esta información la organizaría en un documento estructurado, por componente y con sus respectivas versiones y rutas.

2. Para auditar el estado actual del sistema, utilizaría herramientas como *Lynis* para evaluar la configuración y seguridad, *rpm* y *yum* para obtener detalles de paquetes instalados y dependencias, y *nmap* o *netstat* para examinar puertos y servicios activos. También usaría *crontab -l*, junto con la inspección manual de */etc/cron**, para listar las tareas programadas, y *audit2why* y *audit2allow* para revisar los eventos relacionados con SELinux.

II. Compatibility Analysis

1. Para identificar posibles problemas de compatibilidad entre CentOS 6 y Rocky Linux 8, primero consultaría la documentación oficial de ambos sistemas para conocer diferencias en estructuras de archivos, versiones de paquetes y cambios de comportamiento entre servicios. También usaría rpm para comparar versiones de paquetes actuales y nuevos. Una estrategia efectiva sería montar una máquina virtual con Rocky Linux 8 y replicar el entorno usando contenedores o scripts de automatización para detectar conflictos al iniciar los servicios o ejecutar scripts.

2. Para probar el código PHP 5.6 con PHP 7.4, configuraría un entorno de prueba con PHP 7.4 y desplegaría la aplicación allí. Usaría herramientas como *php7.4 -l* para verificar la sintaxis y *phpcompatinfo* o *PHPCompatibility* (plugin para PHP CodeSniffer) para detectar funciones obsoletas o incompatibles. También realizaría pruebas funcionales manuales y automáticas (si existen suites de pruebas) para identificar errores de ejecución o lógicas rotas.

III. Risk Assessment

1. Los riesgos críticos en esta migración incluyen la posible incompatibilidad de código PHP, fallos en las configuraciones personalizadas como SELinux y reglas de firewall, y pérdida de datos durante la migración de bases de datos. Además, puede haber interrupciones en el servicio si no se realiza una transición cuidadosa o si los nuevos servicios no se comportan como los antiguos.

2. Para planificar una reversión en caso de fallo, implementaría una estrategia de respaldo completo antes de la migración, incluyendo snapshots del servidor, backups de la base de datos (usando *mysqldump* o *mariabackup*) y copias de configuraciones críticas. También mantendría el servidor CentOS 6 operativo y sin cambios hasta verificar que todo funciona correctamente en Rocky Linux 8. Si algo falla, simplemente redirigiría el tráfico de vuelta al servidor original mientras se diagnostican y corrigen los problemas.

Fase 2: Ejecución de la migración

I. Service Migration:

1. Para migrar las configuraciones de Apache de la versión 2.2 a 2.4, primero revisaría los archivos de configuración actuales (*httpd.conf, conf.d/, .htaccess*) y los compararía con los cambios introducidos en Apache 2.4, especialmente en directivas como *Require*, *AccessOrder*, y el manejo de módulos. Usaría la documentación oficial para identificar directivas obsoletas y realizar las modificaciones necesarias. Luego, adaptaría las configuraciones, probándolas en el nuevo entorno con *apachectl configtest* y corrigiendo los errores antes de poner en producción.

2. Para migrar los datos de MySQL 5.1 a MariaDB 10.3, utilizaría mysqldump con las opciones *--single-transaction --routines --triggers* para generar un volcado completo de la base de datos, asegurando compatibilidad con funciones y procedimientos almacenados. Luego importaría este dump en el servidor MariaDB usando *mysql* o *mariadb*. Es importante revisar la codificación de caracteres y ajustar configuraciones del archivo *my.cnf* en caso de diferencias relevantes entre versiones.

II. Application Testing:

1. Para configurar un entorno de staging, levantaría una máquina virtual o contenedor con Rocky Linux 8, instalando las versiones objetivo de Apache, MariaDB y PHP. Clonaría la aplicación, restauraría la base de datos y aplicaría las mismas configuraciones y scripts que se usan en producción. El staging debe tener el mismo entorno de red, permisos y variables de entorno para asegurar pruebas representativas antes del cambio definitivo.

2. Para verificar la compatibilidad del código PHP, emplearía *PHPCompatibility* con PHP *CodeSniffer* para escanear el código y detectar funciones, extensiones o sintaxis incompatibles. Además, usaría *php -l* para verificar la sintaxis de cada archivo y correría pruebas funcionales o unitarias si existen. También sería útil activar el log de

errores de PHP y configurar el entorno en modo desarrollo (*display_errors = On*) para detectar fallos en tiempo de ejecución.

III. User and Permission Migration:

1. Para asegurar que los permisos y la propiedad de los archivos se mantengan, usaría el comando *rsync -a* (modo archivo, que conserva permisos, propietarios, tiempos y enlaces simbólicos) al copiar datos entre servidores. Antes de transferir, me aseguraría de que los UID y GID coincidan entre los sistemas origen y destino para evitar discrepancias en la asignación de propietarios.

2. Para migrar cuentas de usuarios y grupos locales, extraería los datos de los archivos */etc/passwd*, */etc/shadow*, y */etc/group*, filtrando solo los usuarios relevantes (por ejemplo, los no del sistema). Luego usaría scripts o herramientas como *getent* y *awk* para exportarlos e importarlos en el nuevo sistema. También verificaría que los home directories, shells y permisos estén correctamente restaurados, y que los UID y GID no entren en conflicto con los del nuevo entorno.

Fase 3: Post-Migración

I. Validation:

1. Para verificar que todos los servicios estén funcionando correctamente, comprobaría el estado de los servicios con systemctl status para Apache, MariaDB, y cualquier otro proceso relevante. También revisaría los logs de cada servicio (*/var/log/httpd/*, */var/log/mariadb/*, */var/log/messages*, etc.) en busca de errores. Confirmaría que los puertos necesarios estén abiertos con *ss -tuln* y realizaría pruebas de acceso a la aplicación web para asegurar su funcionalidad completa. Además, verificaría las políticas SELinux con *sestatus* y *audit.log* para asegurar que no estén bloqueando

procesos.

2. Para probar que los cron jobs funcionan correctamente, revisaría su presencia con *crontab -l* y examinaría los archivos en */etc/cron**. Luego, forzaría la ejecución manual de algunos cron jobs para confirmar que se ejecutan sin errores. También activaría el logging de cron (si no lo está) revisando */var/log/cron* para confirmar que las tareas se están ejecutando en el momento programado.

II. Performance Benchmarking:

1. Para comparar el rendimiento entre los sistemas viejo y nuevo, ejecutaría pruebas de carga y respuesta utilizando herramientas como *ApacheBench (ab)*, *Siege*, o *wrk* para medir el tiempo de respuesta del servidor web y consultas a la base de datos. También observaría el uso de CPU, memoria, disco y red con herramientas como *top*, *htop*, *iotop* y *dstat* antes y después de la migración bajo condiciones similares.

2. Documentaría métricas clave como tiempo promedio de respuesta, número de peticiones por segundo, carga del sistema (*load average*), uso de CPU y RAM, latencia de la base de datos, y tiempos de ejecución de scripts programados. Esto permite medir mejoras o detectar cuellos de botella después de la migración.

III. Documentation Update:

1. Se debe actualizar toda la documentación relacionada con versiones de software, rutas de configuración, estructura de directorios, detalles de usuarios y permisos, scripts cron, reglas de firewall, y políticas SELinux. También se debe reflejar cualquier cambio en las dependencias o en los procesos operativos diarios.

2. Para documentar el proceso de migración, elaboraría un informe paso a paso detallando cada fase: evaluación previa, ajustes realizados, comandos utilizados, pruebas efectuadas, incidencias encontradas y cómo se resolvieron. Incluiría

capturas de configuración clave, logs relevantes y fechas del cambio. Este documento serviría como guía futura para migraciones similares o como referencia ante posibles problemas.

Documentación Requerida

I. Lista de Verificación Previa a la Migración:

- **Inventario de todos los servicios y sus dependencias:** Se listaron Apache 2.2, MySQL 5.1, PHP 5.6, scripts personalizados, cron jobs, reglas de firewall y políticas SELinux, junto con sus archivos de configuración y rutas específicas.

- **Informes de verificación de respaldos:** Se realizaron respaldos completos del sistema, incluyendo base de datos (*mysqldump*), archivos de configuración y contenido de aplicaciones, verificados mediante restauración en un entorno de prueba.

- **Matriz de evaluación de riesgos:** Se identificaron riesgos como incompatibilidades de versiones, pérdida de datos, fallos de configuración y tiempos de inactividad; se asignaron niveles de impacto y probabilidad, con medidas preventivas para cada uno.

II. Plan de Migración:

- **Procedimiento paso a paso de la migración:** Instalación de Rocky Linux 8, configuración de servicios equivalentes (Apache 2.4, MariaDB 10.3, PHP 7.4), migración de datos, ajustes de configuraciones y pruebas en entorno de staging.

- **Procedimiento de reversión en caso de falla:** Conservación del servidor original en línea, backups listos para restauración, posibilidad de redirigir tráfico al sistema antiguo si se detectan fallos críticos.

- **Metodología de pruebas y validación:** Validación manual y automatizada de funcionalidades clave, pruebas de rendimiento con herramientas como ab y revisión de logs de errores tras cada etapa.

III. Informe Posterior a la Migración:

- **Problemas encontrados y cómo se resolvieron:** Se detectaron funciones PHP obsoletas que fueron actualizadas, conflictos de configuración en Apache que se adaptaron a la sintaxis 2.4 y se ajustaron políticas SELinux que bloqueaban ciertos scripts.

- **Datos comparativos de rendimiento entre sistemas:** Se observó una mejora en el tiempo de respuesta web y en la eficiencia del uso de recursos, especialmente en el manejo de múltiples conexiones a base de datos.

- **Documentación del sistema actualizada y centralizada:** Se actualizó toda la documentación técnica en un repositorio interno, incluyendo versiones, rutas, usuarios, permisos, configuraciones personalizadas y procesos recurrentes.