*Facultad de Ingeniería y Sistemas*

*Ingeniería en Sistemas Computacionales*

*MATERIA:*

*Prog. De Computadoras II*

*Tema:*

*Versión de Java*

*Facilitadora:*

*Ing. Abdel G. Martínez L.*

*Estudiante:*

*Héctor Mojica*

*FECHA DE ENTREGA:*

*08 de Octubre de 2015*

Tarea #2

1- ¿Cuál es la última versión de Java?

**R**: La versión más reciente es Java: 8.0, la cual tuvo una actualización reciente en el paquete de actualización **Java 8 Update 60 (8u60)**

2- Características nuevas de la versión

* Datos IANA 2015e  
  JDK 8u60 contiene datos de zona horaria IANA versión 2015e. Para obtener más información, consulte [Versiones de datos de zona horaria en el software de JRE](http://www.oracle.com/technetwork/java/javase/tzdata-versions-138805.html).
* **Corrección de bug:** dns\_lookup\_realm debe estar definido en false por defecto  
  El valor de dns\_lookup\_realm del archivo Kerberos' krb5.conf es por defecto false. Consulte[8080637](http://bugs.java.com/view_bug.do?bug_id=8080637).
* **Corrección de bug:** Desactivar los conjuntos de cifrado RC4  
  Los conjuntos de cifrado TLS basados en RC4 (por ejemplo, TLS\_RSA\_WITH\_RC4\_128\_SHA) ya no son seguros, por lo que no deben seguir usándose (consulte RFC 7465). Como consecuencia, los conjuntos de cifrado TLS basados en RC4 se han desactivado por defecto en la implantación de Oracle JSSE agregando "RC4" a la propiedad de seguridad "jdk.tls.disabledAlgorithms" y eliminándolos de la lista de conjuntos de cifrado activados por defecto. Estos conjuntos de cifrado pueden reactivarse eliminando "RC4" de la propiedad de seguridad "jdk.tls.disabledAlgorithms" en el archivo java.security o llamando de forma dinámica a Security.setProperty() y también volviendo a agregarlos a la lista de conjuntos de cifrado activados con los métodos SSLSocket/SSLEngine.setEnabledCipherSuites(). También puede utilizar la opción de línea de comando -Djava.security.properties para sobrescribir la propiedad de seguridadjdk.tls.disabledAlgorithms. Por ejemplo:  
  java -Djava.security.properties=my.java.security ...   
  donde my.java.security es un archivo que contiene la propiedad sin RC4:  
  jdk.tls.disabledAlgorithms=SSLv3   
  Incluso con esta opción definida desde la línea de comandos, deben volver a agregarse los conjuntos de cifrado basados en RC4 a la lista de conjuntos de cifrado activados mediante los métodos SSLSocket/SSLEngine.setEnabledCipherSuites(). Consulte [8076221](http://bugs.java.com/view_bug.do?bug_id=8076221).
* **Corrección de bug:** Soportar la detección de tipo de almacén de claves para almacenes de claves JKS y PKCS12  
  Modo de compatibilidad de almacén de claves: Para mejorar la interoperabilidad, el tipo de almacén de claves de Java JKS ahora soporta el modo de compatibilidad de almacén de claves por defecto. Este modo permite a los almacenes de claves de JKS acceder a los formatos de archivo JKS y PKCS12. Para desactivar el modo de compatibilidad de almacén de claves, defina la propiedad de seguridad keystore.type.compat en el valor de cadena false. Consulte[8062552](http://bugs.java.com/view_bug.do?bug_id=8062552).
* **Corrección de bug:** Métodos de supervisión Unsafe anticuados en la versión JDK 8u  
  Los métodos monitorEnter, monitorExit y tryMonitorEnter en sun.misc.Unsafe se han marcado como anticuados en JDK 8u60 y se eliminarán en una futura versión. Estos métodos no se utilizan en el JDK en sí y rara vez se usan fuera del JDK. Consulte [8069302](http://bugs.java.com/view_bug.do?bug_id=8069302).
* **Corrección de bug:** Extraer la grabación de JFR del archivo principal mediante SA  
  DumpJFR es una herramienta basada en agente de capacidad de servicio que puede usarse para extraer datos de Java Flight Recorder (JFR) de los archivos principales y los procesos de Hotspot activos. Se puede usar DumpJFR mediante uno de los métodos siguientes:
  + Añada DumpJFR a un proceso activo:  
      
    java -cp $JAVA\_HOME/lib/sa-jdi.jar sun.jvm.hotspot.tools.DumpJFR <pid>
  + Añada DumpJFR a un archivo principal:  
      
    java -cp $JAVA\_HOME/lib/sa-jdi.jar sun.jvm.hotspot.tools.DumpJFR <java> <core>

La herramienta DumpJFR vuelca los datos de JFR en un archivo denominado recording.jfr en la carpeta de trabajo actual. Consulte 8065301 (no público).

* **Corrección de bug:** Las variables locales denominadas 'enum' provocan fallos falsos del compilador  
  El analizador javac analiza de forma incorrecta variables locales con el nombre 'enum'; esto provoca fallos falsos cuando un programa con dichas variables locales se compila con un indicador 'source' que corresponde a una versión en la que la construcción enum no está disponible (como '-source 1.4'). Consulte [8069181](http://bugs.java.com/view_bug.do?bug_id=8069181).

**Java Development Kit para ARM versión 8u60**

Esta versión incluye Java Development Kit para ARM versión 8u60 (JDK 8u60 para ARM). Para obtener información de soporte de dispositivos ARM, consulte la página [Descargas de JDK para ARM](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-arm-downloads-2187472.html). Para obtener una lista de los requisitos del sistema, instrucciones de instalación y consejos de resolución de problemas, consulte la página [Instrucciones de instalación](http://docs.oracle.com/javase/8/docs/technotes/guides/install/index.html).

**Limitación:** El soporte de Native Memory Tracking está limitado en JDK para ARM. La opción de línea de comando de Java XX:NativeMemoryTracking=detail no está soportada para destinos de ARM (se muestra un mensaje de error al usuario). En su lugar utilice la siguiente opción:  
XX:NativeMemoryTracking=summary

**Actualizaciones en la documentación por las mejoras de Nashorn**

JDK 8u60 incluye nuevas mejoras de Nashorn. Como consecuencia, deben leerse los siguientes cambios en la documentación junto con la documentación de Nashorn existente:

* Adición: En la [sección anterior](http://docs.oracle.com/javase/8/docs/technotes/guides/scripting/prog_guide/javascript.html#A1147390), mencionamos que todos los objetos JavaScript expuestos a API Java implantan la interfaz java.util.Map. Esto se aplica también a las matrices de JavaScript. Sin embargo, a menudo este no es el comportamiento que se espera o desea cuando el código Java espera objetos analizados por JSON. Las bibliotecas de Java que manipulan objetos analizados por JSON normalmente esperan matrices para exponer en su lugar la interfazjava.util.List. Si necesita exponer sus objetos de JavaScript para exponer las matrices como listas y no como mapas, puede utilizar la función Java.asJSONCompatible(obj), donde obj es la raíz del árbol de objetos de JSON.
* Corrección: La precaución mencionada al final de la sección [Asignación de tipos de datos](http://docs.oracle.com/javase/8/docs/technotes/guides/scripting/prog_guide/javascript.html#A1147390) ya no es aplicable. Nashorn garantiza que las cadenas de JavaScript internas se convierten ajava.lang.String al exponerse de forma externa.
* Corrección: La afirmación de la sección [Asignación de tipos de datos](http://docs.oracle.com/javase/8/docs/technotes/guides/scripting/prog_guide/javascript.html#A1147390), que menciona "Por ejemplo, las matrices deben convertirse de forma explícita,..." no es correcta. Las matrices se convierten de forma automática a tipos de matriz Java, como java.util.List, java.util.Collection,java.util.Queue y java.util.Deque, etc.

**Cambios en el juego de reglas de despliegue v1.2**

JDK 8u60 implanta el juego de reglas de despliegue (DRS) 1.2, que incluye los siguientes cambios:

* Agregue el elemento "checksum" como subelemento de "id", lo que permite que los jar no firmados sean identificados por el total de control SHA-256 del formato sin comprimir de un jar:
  + El elemento "checksum" se hará coincidir solo con jar no firmados, y el hash proporcionado se comparará solo con el formato no comprimido del jar.
  + El elemento "checksum" (similar al elemento "certificate") tiene dos argumentos"hash" y "algorithm"; sin embargo, a diferencia de lo que ocurre con el elemento"certificate", el único valor soportado para "algorithm" es "SHA-256". Se ignorará cualquier otro valor proporcionado.
* Permite que el elemento "message" se aplique a todos los tipos de regla, donde antes solo se aplicaba a una regla de bloque:
  + En una regla de ejecución, un subelemento de mensaje hará que se muestre un cuadro de diálogo de mensaje mientras que, sin una regla de ejecución, el comportamiento por defecto sería mostrar un cuadro de diálogo de certificado o formato no firmado. El mensaje se mostrará en el cuadro de diálogo de mensaje.
  + En una regla por defecto, el mensaje solo se mostrará si la acción por defecto es el bloqueo. En ese caso, el mensaje se incluirá en el cuadro de diálogo de bloque.
* Repita los bloques "customer" en la consola de Java, los archivos de rastreo y los registros de Java Usage Tracker.
  + Antes de la versión DRS 1.2, los elementos "customer" podían incluirse (con cualquier subelemento) en el archivo ruleset.xml. Tanto este elemento como todos sus subelementos se ignoran. En DRS 1.2, los elementos siguen ignorándose desde el punto de vista funcional. No obstante:  
    - Al analizar el archivo ruleset.xml, todos los bloques "customer" se repetirán en la consola de Java y el archivo de rastreo de despliegue (si la consola y el rastreo están activados).
    - Al utilizar una regla, todos los registros "customer" incluidos en dicha regla se agregarán al registro de Java Usage Tracker (JUT) (si JUT está activado).

Como resultado de los cambios anteriormente indicados, el DTD para DRS 1.2 es:

<!ELEMENT ruleset (rule\*)>

<!ATTRIBUTE ruleset href CDATA #IMPLIED>

<!ATTRIBUTE ruleset version CDATA #REQUIRED>

<!ELEMENT rule (id, action)>

<!ELEMENT id (certificate?) (checksum?) >

<!ATTRIBUTE id title CDATA #IMPLIED>

<!ATTRIBUTE id location CDATA #IMPLIED>

<!ELEMENT certificate EMPTY>

<!ATTLIST certificate algorithm CDATA #IMPLIED>

<!ATTLIST certificate hash CDATA #REQUIRED>

<!ELEMENT checksum EMPTY>

<!ATTLIST checksum algorithm CDATA #IMPLIED>

<!ATTLIST checksum hash CDATA #REQUIRED>

<!ELEMENT action (message?)>

<!ATTRIBUTE permission (run | block | default) #REQUIRED>

<!ATTRIBUTE version CDATA #IMPLIED>

<!ATTRIBUTE force (true|false) "false">

<!ELEMENT message (#PCDATA)>

<!ATTLIST message locale CDATA #IMPLIED>

**Fecha de caducidad de Java**

La fecha de caducidad de 8u60 es el 20 de octubre de 2015. Java caduca cada vez que hay disponible una nueva versión con correcciones a las vulnerabilidades de seguridad. Para los sistemas que no se pueden ejecutar en servidores Oracle, un mecanismo secundario se encargará de caducar esta versión de JRE (versión 8u60) el 20 de noviembre de 2015. Una vez se haya cumplido cualquiera de las condiciones (la nueva versión esté disponible o se haya alcanzado la fecha de caducidad) Java enviará mensajes de advertencia y recordatorios sobre la nueva versión.

**Correcciones de bugs**

Para acceder a una lista de las correcciones de bugs que incluye esta versión, consulte la página[Correcciones de bugs de JDK 8u60](http://www.oracle.com/technetwork/java/javase/2col/8u60-bugfixes-2620228.html).

» [Notas técnicas sobre la versión 8u60](http://www.oracle.com/technetwork/java/javase/8u60-relnotes-2620227.html)

**Java 8 Update 51 (8u51)**

**Características principales de la versión**

* Datos IANA 2015d  
  JDK 8u51 contiene datos de zona horaria IANA versión 2015d. Para obtener más información, consulte [Versiones de datos de zona horaria en el software de JRE](http://www.oracle.com/technetwork/java/javase/tzdata-versions-138805.html).
* **Corrección del bug:** Agregar nuevas raíces de Comodo a las CA raíz  
  Se han agregado cuatro certificados raíz nuevos para Commodo:
  1. Autoridad de certificación COMODO ECC  
     alias: comodoeccca  
     DN: CN=COMODO ECC Certification Authority, O=COMODO CA Limited, L=Salford, ST=Greater Manchester, C=GB
  2. Autoridad de certificación COMODO RSA  
     alias: comodorsaca  
     DN: CN=COMODO RSA Certification Authority, O=COMODO CA Limited, L=Salford, ST=Greater Manchester, C=GB
  3. Autoridad de certificación USERTrust ECC  
     alias: usertrusteccca  
     DN: CN=USERTrust ECC Certification Authority, O=The USERTRUST Network, L=Jersey City, ST=New Jersey, C=US
  4. Autoridad de certificación USERTrust RSA  
     alias: usertrustrsaca  
     DN: CN=USERTrust RSA Certification Authority, O=The USERTRUST Network, L=Jersey City, ST=New Jersey, C=US

Consulte JDK-8077997 (no público).

* **Corrección del bug:** Agregar nuevas raíces de GlobalSign a las CA raíz  
  Se han agregado dos certificados raíz nuevos para GlobalSign:
  1. CA raíz de GlobalSign ECC - R4  
     alias: globalsigneccrootcar4  
     DN: CN=GlobalSign, O=GlobalSign, OU=GlobalSign ECC Root CA - R4
  2. CA raíz de GlobalSign ECC - R4  
     alias: globalsigneccrootcar5  
     DN: CN=GlobalSign, O=GlobalSign, OU=GlobalSign ECC Root CA - R5

Consulte JDK-8077995 (no público).

* **Corrección del bug:** Agregar Actalis a CA raíz  
  Se ha agregado un nuevo certificado raíz:  
  CA raíz de autenticación Actalis  
  alias: actalisauthenticationrootca  
  DN: CN=Actalis Authentication Root CA, O=Actalis S.p.A./03358520967, L=Milan, C=IT   
  Consulte JDK-8077903 (no público).
* **Corrección del bug:** Agregar nueva raíz de Entrust ECC  
  Se ha agregado un nuevo certificado raíz:  
  Autoridad de certificación raíz de Entrust - EC1  
  alias: entrustrootcaec1  
  DN: CN=Entrust Root Certification Authority - EC1, OU="(c) 2012 Entrust, Inc. - for authorized use only", OU=See www.entrust.net/legal-terms, O="Entrust, Inc.", C=US   
  Consulte JDK-8073286 (no público).
* **Corrección del bug:**: Eliminar las raíces de Valicert Class 1 y 2 Policy  
  Se han eliminado dos certificados raíz con claves de 1024 bits:
  1. Autoridad de validación de políticas ValiCert Class 1  
     alias: secomvalicertclass1ca  
     DN: EMAILADDRESS=info@valicert.com, CN=http://www.valicert.com/, OU=ValiCert Class 1 Policy Validation Authority, O="ValiCert, Inc.", L=ValiCert Validation Network
  2. Autoridad de validación de políticas ValiCert Class 2  
     alias: valicertclass2ca  
     DN: EMAILADDRESS=info@valicert.com, CN=http://www.valicert.com/, OU=ValiCert Class 2 Policy Validation Authority, O="ValiCert, Inc.", L=ValiCert Validation Network

Consulte JDK-8077886 (no público).

* **Corrección del bug:** Eliminar las raíces de Thawte antiguas  
  Se han eliminado dos certificados raíz con claves de 1024 bits:
  1. CA de servidor Thawte  
     alias: thawteserverca  
     DN: EMAILADDRESS=server-certs@thawte.com, CN=Thawte Server CA, OU=Certification Services Division, O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA
  2. CA de Thawte Personal Freemail  
     alias: thawtepersonalfreemailca  
     DN: EMAILADDRESS=personal-freemail@thawte.com, CN=Thawte Personal Freemail CA, OU=Certification Services Division, O=Thawte Consulting, L=Cape Town, ST=Western Cape, C=ZA

Consulte JDK-8074423 (no público).

* **Corrección del bug:** Eliminar más raíces de Verisign, Equifax y Thawte antiguas  
  Se han eliminado cinco certificados raíz con claves de 1024 bits:
  1. Autoridad de certificación principal pública de Verisign Class 3 - G2  
     alias: verisignclass3g2ca DN: OU=VeriSign Trust Network, OU="(c) 1998 VeriSign, Inc. - For authorized use only", OU=Class 3 Public Primary Certification Authority - G2, O="VeriSign, Inc.", C=US
  2. CA de servidor Thawte Premium  
     alias: thawtepremiumserverca  
     DN: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA, OU=Certification Services Division, O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA
  3. Autoridad de certificación de Equifax Secure  
     alias: equifaxsecureca  
     DN: OU=Equifax Secure Certificate Authority, O=Equifax, C=US
  4. CA-1 de Equifax Secure eBusiness  
     alias: equifaxsecureebusinessca1  
     DN: CN=Equifax Secure eBusiness CA-1, O=Equifax Secure Inc., C=US
  5. CA-1 de Equifax Secure Global eBusiness  
     alias: equifaxsecureglobalebusinessca1  
     DN: CN=Equifax Secure eBusiness CA-1, O=Equifax Secure Inc., C=US

Consulte JDK-8076202 (no público).

* **Corrección del bug:** Eliminar las raíces de CA de TrustCenter antiguas  
  Se han eliminado tres certificados raíz:
  1. CA I de TC TrustCenter Universal   
     alias: trustcenteruniversalcai  
     DN: CN=TC TrustCenter Universal CA I, OU=TC TrustCenter Universal CA, O=TC TrustCenter GmbH, C=DE
  2. CA II de TC TrustCenter Class 2   
     alias: trustcenterclass2caii  
     DN: CN=TC TrustCenter Class 2 CA II, OU=TC TrustCenter Class 2 CA, O=TC TrustCenter GmbH, C=DE
  3. CA II de TC TrustCenter Class 4   
     alias: trustcenterclass4caii  
     DN: CN=TC TrustCenter Class 4 CA II, OU=TC TrustCenter Class 4 CA, O=TC TrustCenter GmbH, C=DE

Consulte JDK-8072958 (no público).

* **Corrección del bug:** RC4 en proveedor SunJSSE obsoleto   
  RC4 se considera un cifrado débil. Los servidores no deben seleccionar RC4 a menos que no haya otro cliente más fuerte en las suites de cifrado solicitadas. Se ha agregado una nueva propiedad de seguridad, jdk.tls.legacyAlgorithms, para definir los algoritmos antiguos en la implantación de Oracle JSSE. Se han agregado los algoritmos relacionados con RC4 a la lista de algoritmos antiguos. Consulte JDK-8074006 (no público).
* **Corrección del bug:** Prohibir suites de cifrado RC4  
  RC4 es ahora un cifrado no seguro. Se han eliminado las suites de cifrado RC4 de las listas de suites de cifrado activadas por defecto en el cliente y del servidor para la implantación de Oracle JSSE. Estas suites de cifrado aún se pueden activar con los métodos[SSLEngine.setEnabledCipherSuites()](http://docs.oracle.com/javase/8/docs/api/javax/net/ssl/SSLEngine.html#setEnabledCipherSuites-java.lang.String:A-) y [SSLSocket.setEnabledCipherSuites()](http://docs.oracle.com/javase/8/docs/api/javax/net/ssl/SSLSocket.html#setEnabledCipherSuites-java.lang.String:A-). Consulte JDK-8077109 (no público).
* **Corrección del error:** Comprobación de certificación mejorada  
  Con esta corrección, la identificación de punto final de JSSE no realiza la consulta de nombre inversa para las direcciones IP por defecto en JDK. Si una aplicación necesita realizar consulta de nombre inversa para las direcciones IP raíz en las conexiones SSL/TLS y encuentra un problema de compatibilidad de identificación de punto final, se puede utilizar la propiedad del sistema "jdk.tls.trustNameService" en la consulta de nombre inversa. Tenga en cuenta que si el servicio de nombres no es de confianza, al activar la consulta de nombre inversa se pueden producir ataques de MITM. Consulte JDK-8067695 (no público).

**Fecha de caducidad de Java**

La fecha de caducidad de 8u51 es el 20 de octubre de 2015. Java caduca cada vez que hay disponible una nueva versión con correcciones a las vulnerabilidades de seguridad. Para los sistemas que no se pueden ejecutar en servidores Oracle, un mecanismo secundario se encargará de caducar esta versión de JRE (versión 8u51) el 20 de noviembre de 2015. Una vez se haya cumplido cualquiera de las condiciones (la nueva versión esté disponible o se haya alcanzado la fecha de caducidad) Java enviará mensajes de advertencia y recordatorios sobre la nueva versión.

**Correcciones de bugs**

Esta versión incluye correcciones para vulnerabilidades de seguridad. Para obtener más información, consulte el [Asesor de actualización de parche crítico de Oracle Java SE](http://www.oracle.com/technetwork/topics/security/cpujul2015-2367936.html#AppendixJAVA).

Para acceder a una lista de las correcciones de bugs que incluye esta versión, consulte la página[Correcciones de bugs de JDK 8u51](http://www.oracle.com/technetwork/java/javase/2col/8u51-bugfixes-2587594.html).

3-Cuando se espera la Próxima versión de Java.

R: Se espera la próxima actualización de **Java** para el 2016 con el nombre de **Java SE (Standard Edition) 9**