1. **De las siguientes opciones, ¿Cuál es una característica de un protocolo TCP?**
2. **El consumidor no espera cuando no hay datos disponibles.**
3. **El productor espera cuando hay espacio en la cola.**
4. Permite multiplexar los datos, es decir, que la información que viene de diferentes fuentes en la misma línea pueda circular simultáneamente.
5. **A Y B**
6. **B Y C**

**Explicacion:**

**Las principales características del protocolo TCP son las siguientes:**

1. **TCP permite colocar los datagramas nuevamente en orden cuando vienen del protocolo IP.**
2. **TCP permite que el monitoreo del flujo de los datos y así evita la saturación de la red.**
3. **TCP permite que los datos se formen en segmentos de longitud variada para "entregarlos" al protocolo IP.**
4. **TCP permite multiplexar los datos, es decir, que la información que viene de diferentes fuentes (por ejemplo, aplicaciones) en la misma línea pueda circular simultáneamente.**
5. **Por último, TCP permite comenzar y finalizar la comunicación amablemente.**
6. **Verdadero o falso**

**En los mensajes permitidos, cuando el emisor no recibe el acuso de recibo en el plazo permitido, el mensaje se pierde y no vuelve a ser retrasmitido**

1. **Verdadero**
2. **Falso**

**Explicación:**

**En los mensajes permitidos, si el emisor no recibe el acuso de recibo dentro del plazo determinado volverá a transmitir el mensaje.**

1. **¿Cuáles son los aspectos de la comunicación de streams?**
2. **Bloqueo, Mensajes permitidos.**
3. **Concordancia de ítems de datos.**
4. **Hilos, tamaño del mensaje.**
5. **Hilos, concordancia de ítems de datos.**
6. **Control de flujo, destinos de los mensajes.**

**Explicación:**

**Los aspectos de la comunicación de streams.**

**Hilos**: Cuando un servidor acepta una conexión, generalmente crea un

**Bloqueo**: Los datos escritos en el stream se almacenan en un búfer

**Concordancia de ítems de datos**: Los procesos necesitan estar de acuerdo con el tipo y cantidad de datos transmitidos (envía ***int,*** ***double***, recibe ***int, double***). En el socket destino. El proceso que lee, intentará extraer los datos de la cola o se bloqueará hasta recibir datos. El proceso que escribe datos en el stream será bloqueado por el mecanismo de control si el socket emisor intenta enviar más datos de los permitidos. Nuevo hilo para ese cliente, esto evita que el servidor pueda bloquearse a la espera de mensajes sin afectar a los otros clientes.

1. **Según la siguiente definición, como se define a un checksum.**

Es una [función *hash*](https://es.wikipedia.org/wiki/Funci%C3%B3n_hash) que tiene como propósito principal detectar cambios accidentales en una secuencia de datos para proteger la integridad de estos, verificando que no haya discrepancias entre los valores obtenidos al hacer una comprobación inicial y otra final tras la transmisión.

La idea es que se transmita el dato junto con su valor *hash*, de esta forma el receptor puede calcular dicho valor y compararlo así con el valor *hash* recibido. Si hay una discrepancia se pueden rechazar los datos o pedir una retransmisión.

1. **Verdadero**
2. **Falso**

**Explicación:**

Una **suma de verificación**, ( también llamada **suma de chequeo** o **checksum**), en [telecomunicación](https://es.wikipedia.org/wiki/Telecomunicaci%C3%B3n) e [informática](https://es.wikipedia.org/wiki/Inform%C3%A1tica), es una [función hash](https://es.wikipedia.org/wiki/Funci%C3%B3n_hash) que tiene como propósito principal detectar cambios accidentales en una secuencia de datos para proteger la integridad de estos, verificando que no haya discrepancias entre los valores obtenidos al hacer una comprobación inicial y otra final tras la transmisión. La idea es que se transmita el dato junto con su valor hash, de esta forma el receptor puede calcular dicho valor y compararlo así con el valor hash recibido. Si hay una discrepancia se pueden rechazar los datos o pedir una retransmisión.

Esto es empleado para comunicaciones ([Internet](https://es.wikipedia.org/wiki/Internet), comunicación de dispositivos, etc.) y almacenamiento de datos ([archivos comprimidos](https://es.wikipedia.org/wiki/Compresi%C3%B3n_de_datos), [discos portátiles](https://es.wikipedia.org/wiki/Disco_duro_port%C3%A1til), etc.).

Normalmente aumentar la capacidad de detectar más tipos de error aumenta la complejidad del algoritmo y el coste, pues aumenta las necesidades de proceso de éste. Sin embargo, proporciona medios de detectar errores en el código de forma fiable.

1. **¿Qué es un bit de paridad?**
2. Un **bit de paridad** es un [dígito binario](https://es.wikipedia.org/wiki/Bit) que indica si el número de bits con un valor de *1* en un conjunto de bits es [par](https://es.wikipedia.org/wiki/N%C3%BAmeros_pares_e_impares) o [impar](https://es.wikipedia.org/wiki/N%C3%BAmeros_pares_e_impares).
3. Los bits de paridad conforman el método de [detección de errores](https://es.wikipedia.org/wiki/Detecci%C3%B3n_de_errores) más simple.
4. Es un parámetro opcional que se utiliza en las comunicaciones serie para determinar si el dispositivo remoto está recibiendo correctamente el carácter de datos que se transmite.
5. **A,B,C**
6. B,C

**Explicación**

Un **bit de paridad** es un [dígito binario](https://es.wikipedia.org/wiki/Bit) que indica si el número de bits con un valor de *1* en un conjunto de bits es [par](https://es.wikipedia.org/wiki/N%C3%BAmeros_pares_e_impares) o [impar](https://es.wikipedia.org/wiki/N%C3%BAmeros_pares_e_impares). Los bits de paridad conforman el método de [detección de errores](https://es.wikipedia.org/wiki/Detecci%C3%B3n_de_errores) más simple.

La paridad par es un caso especial del [control de redundancia cíclica](https://es.wikipedia.org/wiki/Control_de_redundancia_c%C3%ADclica) (CRC), donde el bit de CRC se genera por el polinomio *x*+1.

Nótese que este método detecta los errores, pero no los corrige (salvo en el caso de que la palabra transmitida sea de tamaño 1 bit). Existen dos variantes de este método, bit de paridad par y bit de paridad impar:

En el caso de la paridad par, se cuentan el número de unos. Si el total es impar, el bit de paridad se establece en uno y por tanto la suma del total anterior con este bit de paridad, daría par. Si el conteo de bits uno es par, entonces el bit de paridad(par) se deja en 0, pues ya es par.

En el caso de la paridad impar, la situación es la contraria. Se suman los bits cuyo valor es uno, si da un número impar de bits, entonces el bit de paridad(impar) es cero. Y si la suma de los bits cuyo valor es uno es par, entonces el bit de paridad(impar) se establece en uno, haciendo impar la cuenta total de bits uno.

1. **Cuáles son los pasos para establecer una conexión TCP por parte del servidor.**

**Crear un Socket especificando la dirección y puerto del servidor.**

**Leer y escribir datos usando el stream asociado al socket.**

El servidor acepta una conexión y crea un nuevo stream para la comunicación con el cliente, manteniendo el socket original para escuchar conexiones entrantes. Un par de sockets en el cliente y el servidor son conectados por un par de streams (uno en cada dirección).

Esperar por peticiones de conexión por parte clientes.

Crear un socket asignado a un puerto para escuchar y mantiene una cola de peticiones de conexión).

Crear un socket asignado a un puerto para escuchar.

Esperar por peticiones de conexión por parte clientes (este socket mantiene la cola de peticiones de conexión).

El servidor acepta una conexión y crea un nuevo stream para la comunicación con el cliente, manteniendo el socket original para escuchar conexiones entrantes. Un par de sockets en el cliente y el servidor son conectados por un par de streams (uno en cada dirección).

**Explicación**

**Servidor**

Crear un socket asignado a un puerto para escuchar.

Esperar por peticiones de conexión por parte clientes (este socket mantiene la cola de peticiones de conexión).

El servidor acepta una conexión y crea un nuevo stream para la comunicación con el cliente, manteniendo el socket original para escuchar conexiones entrantes. Un par de sockets en el cliente y el servidor son conectados por un par de streams (uno en cada dirección).

**Cliente**

Crear un Socket especificando la dirección y puerto del servidor.

Leer y escribir datos usando el stream asociado al socket.

1. **¿Qué es Stream Control Transmission Protocol?**
2. Stream Control Transmission Protocol (SCTP) es un protocolo de comunicación de capa de transporte.
3. SCTP es una alternativa a los protocolos de transporte TCP y UDP pues provee confiabilidad, control de flujo y secuenciación como TCP.
4. SCTP opcionalmente permite el envío de mensajes fuera de orden y a diferencia de TCP, SCTP es un protocolo orientado al mensaje (similar al envío de datagramas UDP).
5. **A,B,C.**
6. Ninguna de las anteriores.

**Explicación:**

Stream Control Transmission Protocol (SCTP) es un protocolo de comunicación de capa de transporte que fue definido por el grupo SIGTRAN de IETF en el año 2000.

SCTP es una alternativa a los protocolos de transporte TCP y UDP pues provee confiabilidad, control de flujo y secuenciación como TCP. Sin embargo, SCTP opcionalmente permite el envío de mensajes fuera de orden y a diferencia de TCP, SCTP es un protocolo orientado al mensaje (similar al envío de datagramas UDP).

Las ventajas de SCTP son:

1. Capacidad de Multihoming, en la cual uno (o dos) de los extremos de una asociación (conexión) pueden tener más de una dirección IP. Esto permite reaccionar en forma transparente ante fallos en la red.
2. Entrega de los datos en trozos que forman parte de flujos independientes y paralelos —eliminando así el problema de head of the line blocking que sufre TCP.
3. Es capaz de seleccionar y monitorizar caminos, seleccionando un camino "primario" y verificando constantemente la conectividad de cada uno de los caminos alternativos.
4. Mecanismos de validación y asentimiento como protección ante ataques por inundación, proveyendo notificación de trozos de datos duplicados o perdidos.
5. **¿Cuáles son las alternativas para la representación externa de datos y empaquetado?**
6. **Representación de datos CORBA**

**Socialización de objetos Java.**

**XML**

**JSON**

1. **Representación externa de datos.**

**Empaquetado o marshalling.**

**Desempaquetado o Unmarshalling.**

1. **Conversión de datos a un formato externo acordado antes de transmitir.**

**Transmitir datos en un formato que el emisor indicado en formato utilizado por el receptor lo convertirá si es necesario.**

1. **Empaquetado de datos**

**IDL**

**Escribiendo una aplicación en CORBA**

1. **Ninguna de las anteriores**

**Explicación**

Existen varias alternativas para la representación externa de datos y el empaquetado, entre ellas está:

La representación común de datos de CORBA

1. Serialización de objetos de Java
2. XML (eXtensible Markup Language)
3. JSON

**Representación externa de datos:** Es el estándar acordado para la representación de estructuras de datos y valores primitivos.

**Empaquetado o marshalling:**

Traducir las estructuras de datos y los valores primitivos en una representación externa de datos.

**Desempaquetado o Unmarshalling:** Generar los valores primitivos desde la representación externa de datos y reconstruir las estructuras de datos.

1. **¿Cuál es la mejor definición de CORBA CDR?**
2. CORBA CDR es la representación externa de datos (Common Object Request Broker Architecture).
3. Common Object Request Broker Architecture (CORBA) es un estándar definido por Object Management Group (OMG) que permite que diversos componentes de software escritos en múltiples lenguajes de programación y que corren en diferentes computadoras, puedan trabajar juntos; es decir, facilita el desarrollo de aplicaciones distribuidas en entornos heterogéneos.
4. **Cuales con los componente s de CORBA CDR**
5. **Intermediario para solicitudes de objetos (ORB)**

**Servidor CORBA**

**Servicio de nombres**

**Nodo CORBA Request**

1. **Factory**

**Bank**

**Correo**

**StockExchange**

**Market**

**Calculator**

1. **cpponly**

**ID**

**Init**

**localonly**

**localonly abstract**

**Prefijo**

**version**

1. **A, B**
2. **Ninguna de las anteriores.**

**Explicación**

CORBA incluye cuatro componentes:

1. Intermediario para solicitudes de objetos (ORB)

El intermediario para solicitudes de objetos (ORB) maneja la comunicación, ordenación y desordenación de parámetros, de modo que el manejo de parámetros es transparente para aplicaciones de cliente y servidor CORBA.

1. Servidor CORBA

El servidor CORBA crea objetos CORBA y los inicializa con un ORB. El servidor coloca las referencias a los objetos CORBA dentro de un servicio de denominación de modo que los clientes puedan acceder a los mismos.

1. Servicio de nombres

El servicio de denominación mantiene referencias a objetos CORBA.

1. Nodo CORBARequest

El nodo CORBARequest actúa como cliente CORBA.}

1. **¿Qué es RMI?**
2. **RMI es un paquete de JAVA que permite manejar objetos de manera remota, para utilizar los recursos de un servidor de manera transparente para el usuario local.**
3. **RMI es la transferencia sincrónica de datos y control entre dos partes de un programa distribuido a través de espacios de direcciones disjuntas.**
4. **Un RMI es, en el contexto del testeo del software, un trozo de código usado como sustituto de alguna otra funcionalidad.**
5. **A,B**
6. **B,C**
7. **¿Cuál de las siguientes opciones es una diferencia notable entre JAVA RMI Y JAVA?**
8. **Una referencia a un objeto, puede ser pasada como argumento o retornado como resultado de cualquier invocación (local o remoto).**
9. **El operador instanceof puede ser utilizado para testear la interface remota soportada por el objeto remoto.**
10. **Los clientes de objetos remotos interactúan con interfaces remotas, nunca con la implementación de esas interfaces.**
11. **Un objeto remoto puede ser invocado sobre cualquier set de interfaces remotas soportadas por la implementacion, usando la sintaxis de "castear", vale decir: InterfaceRemota irem = (InterfaceRemota) new ClaseRemota().**
12. **Ninguna de las anteriores**

**Explicación**

**Diferencias**

1. **Los clientes de objetos remotos interactúan con interfaces remotas, NUNCA con la implementación de esas interfaces.**
2. **Argumentos no-remotos hacía, y resultados desde, una invocación remota a un método son pasados por copia en vez de por referencia. Esto se debe a que las referencias a objetos son útiles solo dentro de una máquina virtual individual.**
3. **Un objeto remoto es pasado por referencia, no copiando su actual implementación remota.**
4. **En RMI, el cliente debe manejar una mayor cantidad de Excepciones, debido a que aumentan los tipos de fallas al invocar objetos remotos que cuando se trabaja de manera local (seguridad, red, rmiregistry, etc)**
5. **¿Qué es DCOM (Distributed Component Object Model)**
6. **DCOM es la versión JAVA de CORBA, y tal como su nombre lo indica, define interfaces que serán utilizadas en un ambiente CORBA.**
7. **DCOM es un estándar para la comunicación de objetos a través de procedimientos/métodos remotos. Fue creado por un consorcio conocido como OGM (Object Management Group).**
8. **DCOM nace como una extensión de COM, y permite la misma comunicación, pero esta vez acotado a los límites de una red.**
9. **Java DCOM provee su propio ORB, que encarga de traducir los objetos java al estándar.**
10. **Ninguna de las anteriores.**

**Explicación**

DCOM (Distributed Component Object Model)

Originalmente existía lo que se denominaba Component Object Model (COM), que permitía comunicación entre los procesos/objetos, pero estaba acotado a los límites de una sola máquina. DCOM nace como una extensión de COM, y permite la misma comunicación, pero esta vez acotado a los límites de una red.

Es un producto desarrollado por una compañía llamada Microsoft, que utiliza RPC para permitir la comunicación de objetos COM, transparentemente, a través de una red. Estuvo disponible por primera vez en 1995 con el "release" inicial de Windows NT 4.

DCOM, posee las ventajas de los sistemas descritos anteriormente ya que está basado en CORBA para trabajar en ambientes heterogéneos, sin embargo, diferencia de CORBA, quien puede correr en cualquier sistema operativo, DCOM esta implementado solo para Windows**.**

1. **Cuáles son las formas de pasar parámetros en JAVA RMI**
2. **Tipos primitivos**

**Los objetos que no son remotos se pasan usando serializacion de objetos de JAVA.**

**Referencias a objetos que implementan la interfaz remote son pasados como referencias remotas.**

1. **Conversión de datos a un formato externo acordado antes de transmitir.**

**Transmitir datos en un formato que el emisor indicado en formato utilizado por el receptor lo convertirá si es necesario.**

**Empaquetado de datos**

1. **JAVA RMI no es un contexto aceptado en JAVA.**
2. **Tipos primitivos**

**Empaquetado de datos**

**Referencias a objetos que implementan la interfaz remote son pasados como referencias remotas.**

1. **Todas las anteriores.**

**Explicación**

**Existen 3 maneras para pasar los parámetros en Java RMI:**

1. Los tipos primitivos (int, boolean, etc) se pasan por valor.
2. Los objetos que no son remotos se pasan por valor utilizando la serialización de objetos de Java (convierte un objeto a un stream de bytes). Para que un objeto sea serializable, debe implementar la interfaz Serializable.
   1. Si un objeto hace referencias u otros objetos, estos también deben ser serializados. Todos los objetos que pueden ser accedidos a través de ese objeto, son serializados.
   2. Lo mismo ocurre cuando un objeto no remoto es retornado desde una invocación a un método remoto.
   3. Java soporta la integridad referencial, esto signiﬁca que si múltiples referencias a un objeto son pasadas de una JVM a otra, todas estas referencias son tratadas como una misma copia en la JVM receptora.

3. Referencias a objetos que implementan la interfaz remote son pasados como referencias remotas. Se pasa el stub del objeto remoto.

1. **¿Qué nos proporciona una política de seguridad?**
2. **Reglas para compartir recursos.**
3. **Proporciona mecanismos de seguridad.**
4. **Proporciona las bases para la mayoría de mecanismos de seguridad.**
5. **Proporciona reglas para la privacidad e integridad requerida para la información compartida y otros límites para la usabilidad de los recursos compartidos.**
6. **Todas las anteriores.**

**Explicación**

**Una política de seguridad proporciona una regla de la privacidad e integridad requerida para la información compartida y otros límites para la usabilidad de los recursos compartidos.**

**Una política de seguridad es reforzada mediante un mecanismo de seguridad.**

1. **Las amenazas se categorizan en**
2. **Fuga.**

**Alteración.**

**Reenviar.**

1. **Suplantar identidad.**

**Alterar mensajes.**

**Denegación de servicio.**

1. **Espiar**

**Suplantar**

**Alterar mensajes**

1. **Fuga**

**Vandalismo**

**Alteración**

1. **Todas las anteriores**

**Explicación**

**Fuga: La obtención de información por receptores no autorizados.**

**Alteración: La alteración no autorizada de la información.**

**Vandalismo: Interferir con la operación de un sistema sin obtener Ninguna ganancia.**

1. **¿Cuáles son los pasos para cifrar un mensaje?**
2. **Redactar el mensaje**

**El emisor cifra el mensaje con la calve publica del receptor**

**El emisor envía el mensaje cifrado al receptor  a través de internet, ya sea por correo electrónico, mensajería instantánea o cualquier otro medio**

**El receptor recibe el mensaje cifrado y lo descifra con su clave privada**

**El receptor ya puede leer el mensaje original que le mandó el emisor**

1. El emisor envía el mensaje cifrado al receptor  a través de internet, ya sea por correo electrónico, mensajería instantánea o cualquier otro medio

Redactar el mensaje

El emisor cifra el mensaje con la calve publica del receptor

El receptor recibe el mensaje cifrado y lo descifra con su clave privada

El receptor ya puede leer el mensaje original que le mandó el emisor

1. Redactar el mensaje

El emisor cifra el mensaje con la calve publica del receptor

El emisor envía el mensaje cifrado al receptor  a través de internet, ya sea por correo electrónico, mensajería instantánea o cualquier otro medio

El receptor ya puede leer el mensaje original que le mandó el emisor

El receptor recibe el mensaje cifrado y lo descifra con su clave privada

1. **A,B**
2. **Ninguna de las anteriores**

**Explicación**

La criptografía es el uso de 2 claves una clave pública y una clave privada del mismo usuario para enviar un mensaje

1. La Clave pública es utilizada para el  cifrado del mensaje
2. La Clave privada para descifrar el mensaje una vez llegado al receptor

Los pasos para cifrar un mensaje son:

1. Redactar el mensaje
2. El emisor cifra el mensaje con la calve publica del receptor
3. El emisor envía el mensaje cifrado al receptor  a través de internet, ya sea por correo electrónico, mensajería instantánea o cualquier otro medio
4. El receptor recibe el mensaje cifrado y lo descifra con su clave privada
5. El receptor ya puede leer el mensaje original que le mandó el emisor
6. **¿Qué es una firma digital?**
7. Es un esquema matemático que sirve para demostrar la autenticidad de un mensaje digital o de un documento electrónico.
8. Una firma digital da al destinatario seguridad en que el mensaje fue creado por el remitente, y que no fue alterado durante la transmisión.
9. Una firma digital básica pierde la certeza de su validez cuando expira el certificado. Una firma digital avanzada sigue siendo válida sin lugar a dudas durante plazos largos que van más allá de la expiración del certificado.
10. **Una ﬁrma digital sirve como una ﬁrma normal, asocia una identidad a un mensaje. En este caso la identidad es el par de claves pública/privada.**
11. **Todas las anteriores.**

**Explicación:**

Una firma digital es un esquema matemático que sirve para demostrar la autenticidad de un mensaje digital o de un documento electrónico. Una firma digital da al destinatario seguridad en que el mensaje fue creado por el remitente, y que no fue alterado durante la transmisión.

Consiste en un método criptográfico que asocia la identidad de una persona o de un equipo informático al mensaje o documento. En función del tipo de firma, puede, además, asegurar la integridad del documento o mensaje.

La firma digital de un documento es el resultado de aplicar cierto algoritmo matemático, denominado función hash, a su contenido y, seguidamente, aplicar el algoritmo de firma (en el que se emplea una clave privada) al resultado de la operación anterior, generando la firma electrónica o digital. El software de firma digital debe además efectuar varias validaciones, entre las cuales podemos mencionar:

\* Vigencia del certificado digital del firmante,

\* Revocación del certificado digital del firmante,

\* Inclusión de sello de tiempo.

1. **¿Qué es un certificado digital?**
2. **El Certificado Digital es el único medio que permite**garantizar técnica y legalmente la identidad de una persona en Internet. Se trata de un requisito indispensable para que las instituciones puedan ofrecer servicios seguros a través de Internet.
3. El certificado digital permite **la firma electrónica de documentos** El receptor de un documento firmado puede tener la seguridad de que éste es el original y no ha sido manipulado y el autor de la firma electrónica no podrá negar la autoría de esta firma.
4. El certificado de un documento es el resultado de aplicar cierto algoritmo matemático, denominado función hash, a su contenido y, seguidamente, aplicar el algoritmo de firma (en el que se emplea una clave privada) al resultado de la operación anterior, generando la firma electrónica o digital.
5. **A,B**
6. Todas las anteriores

**Explicación**

**El Certificado Digital es el único medio que permite garantizar técnica y legalmente la identidad de una persona en Internet. Se trata de un requisito indispensable para que las instituciones puedan ofrecer servicios seguros a través de Internet. Además:**

**El certificado digital permite la firma electrónica de documentos El receptor de un documento firmado puede tener la seguridad de que éste es el original y no ha sido manipulado y el autor de la firma electrónica no podrá negar la autoría de esta firma.**

**El certificado digital permitecifrar las comunicaciones. Solamente el destinatario de la información podrá acceder al contenido de la misma.**

**En definitiva, la principal ventaja es que disponer de un certificado le ahorrará tiempo y dinero al realizar trámites administrativos en Internet, a cualquier hora y desde cualquier lugar.**

**Un Certificado Digital consta de una pareja de claves criptográficas, una pública y una privada, creadas con un algoritmo matemático, de forma que aquello que se cifra con una de las claves sólo se puede descifrar con su clave pareja.**

**El titular del certificado debe mantener bajo su poder la clave privada, ya que si ésta es sustraída, el sustractor podría suplantar la identidad del titular en la red. En este caso el titular debe revocar el certificado lo antes posible, igual que se anula una tarjeta de crédito sustraída.**

1. **Cuando ya los certificados no son válidos.**

**1. Los certificados digitales son válidos desde su emisión hasta la recepción.**

**2. Los certificados digitales se emiten durante un período indefinido o hasta la muerte del titular.**

**3. Cuando la clave privada ya no es secreta.**

**4.** **El propietario sigue en la misma organización.**

**5. Ninguna de las anteriores.**

**Explicación**

Los certificados digitales pueden caducar o revocarse. Las aplicaciones pueden comprobar si hay certificados revocados utilizando el protocolo OCSP, o listas de revocación de certificados (CRL) en servidores LDAP.

Los certificados digitales se emiten durante un período fijo de tiempo y no son válidos después de su fecha de caducidad. Los certificados también dejan de ser fiables por diferentes motivos, entre ellos:

El propietario ha cambiado a una organización distinta.

La clave privada ya no es secreta.

Las aplicaciones pueden comprobar si un certificado se ha revocado enviando una petición a un respondedor OCSP (Online Certificate Status Protocol - Protocolo de estado de certificados en línea), solamente en sistemas UNIX y Windows. Si lo prefiere, pueden acceder a una CRL en un servidor LDAP. La revocación OCSP y la información de CRL la publica una Autoridad de certificación.

1. **¿Cuál es la mejor definición de Object Request Broker (ORB)?**
2. ORB es la representación externa de datos de (Common Object Request Broker Architecture).
3. Object Request Broker (ORB), es un componente fundamental de la arquitectura CORBA y su misión es facilitar la comunicación entre objetos.
4. Object Request Broker (ORB), es un lenguaje que se utiliza para definir las interfaces entre los componentes de una aplicación y es el elemento que soporta la interoperabilidad de la tecnología.
5. Object Request Broker (ORB), proporciona el lenguaje de definición de intercomunicación entre objetos o sistemas operativos.
6. Todas las anteriores
7. **¿Qué es la fuga de información?**
8. Grupo de datos ya supervisados y ordenados, que sirven para construir un mensaje basado en un cierto fenómeno o ente.
9. Todo elemento o acción capaz de atentar contra la seguridad de la información.
10. Es el incidente que pone en poder de una persona ajena a la organización, información confidencial y que sólo debería estar disponible para integrantes de la misma.
11. Es la probabilidad de ocurrencia de un evento que puede ocasionar un daño potencial a servicios, recursos o sistemas de una empresa.
12. Es un mecanismo de seguridad de prevención y corrección empleado para disminuir las vulnerabilidades
13. **¿Qué es la integridad?**
14. Es la propiedad que busca mantener los datos libres de modificaciones no autorizadas.
15. Es la propiedad que busca la protección de datos y de información intercambiada entre un emisor y uno o más destinatarios frente a terceros.
16. Una política fundamental es la que asegura la entrega del mensaje.
17. Una política fundamental es la que asegura la entrega del mensaje.
18. Una política fundamental es la que asegura la entrega del mensaje.
19. **¿Qué es el criptoanalisis?**
20. Es el estudio del sentido oculto o lo que se esconde en determinados mensajes.
21. Es la alteración de la lengua hablada, mediante una clave concreta y convencional, con el objetivo de que un oyente que no la posea no pueda comprender el mensaje.
22. Es el estudio de los métodos para obtener el sentido de una información cifrada, sin acceso a la información secreta requerida para obtener este sentido normalmente.
23. A,B,C
24. Ninguna de las anteriores.
25. **¿Qué son políticas de seguridad?**
26. La política de seguridad es un conjunto de leyes, reglas y prácticas que regulan la manera de dirigir, proteger y distribuir recursos en una organización para llevar a cabo los objetivos de seguridad informática dentro de la misma.
27. Las políticas de seguridad definen lo que está permitido y lo que está prohibido, permiten definir los procedimientos y herramientas necesarias, expresan el consenso de los “dueños” y permiten adoptar una buena actitud dentro de la organización.
28. Es el conjunto de requisitos definidos por los responsables directos o indirectos de un sistema que indica en términos generales qué está y qué no está permitido en el área de seguridad durante la operación general de dicho sistema.
29. La política de seguridad es un documento de alto nivel que denota el compromiso de la gerencia con la seguridad de la información. Contiene la definición de la seguridad de la información desde el punto de vista de cierta entidad.
30. Todas las anteriores
31. **¿Cuáles son las ventajas de CoﬀeeScript?**
32. Se integra a JQuery de manera sencilla
33. Tiende a introducir una cantidad enorme de fragmentos de código en nuestros sitios web
34. Es legible y entendible
35. **A y B**
36. A y C

**Explicación**

Ventajas

Veamos ahora alguno de los beneficios que nos ofrece CoffeeScript:

* Reduce grandes cantidades de lineas de código en unas pocas
* Es legible y entendible
* Fácil de mantener
* Se integra a JQuery de manera sencilla

1. **¿Qué es Redis?**
   1. Es una solución de código abierto de almacenamiento de datos NoSQL basado en una estructura de Llave-Valor (key-value).
   2. **Redis** es un lenguaje de script muy simple. Orientado a objetos (como [Actionscript 2](http://www.cristalab.com/tutoriales/tutorial-de-actionscript-2.0-en-flash-c21l/), sin clases) y basado en la sintaxis ECMAScript, derivado de C. Fue pensado para crear sencillas interacciones en la web, pero hoy es uno de los lenguajes criticos de las plataformas modernas.
   3. **Redis** es un tipo de **archivo JS** asociado a LXFDVD121 desarrollado por Future Publishing para el Sistema Operativo de Windows. La última versión conocida de Share.js es 1.0.0.0 que se realizó para Windows.
   4. **A y B**
   5. **Ninguna de las anteriores**

**Explicación**

Es una solución de código abierto de almacenamiento de datos NoSQL basado en una estructura de Llave-Valor (key-value). El concepto de este tipo de bases de datos se enfoca en aquellos casos donde una estructura relacional no es necesaria sino más bien un identificador asociado a un valor o un conjunto de valores. Su nombre es un acrónimo de **S**ervidor de **DI**ccionario **RE**moto, puedes encontrar el código fuente del proyecto en su [respositorio de Github](https://github.com/antirez/redis).

Redis es sumamente veloz, esto debido a su sencilla estructura ausente de lógica relacional y su punto más importante, **la base de datos está cargada en memoria**, por ello sus altos niveles de velocidad. No te preocupes, incluso cuando la información está en memoria y en efecto se pedería al apagarse el computador, Redis posee 2 estrategias de persistencia para mantener los datos seguros de los cuales hablaremos en la parte de configuración.

1. **¿Qué es V8 (motor JavaScript)?**
2. **V8** es un motor de [código abierto](https://es.wikipedia.org/wiki/C%C3%B3digo_abierto) para [JavaScript](https://es.wikipedia.org/wiki/JavaScript) creado por [Google](https://es.wikipedia.org/wiki/Google), siendo su [programador](https://es.wikipedia.org/wiki/Programador) jefe Lars Bak.
3. V8 JavaScript está integrado en el navegador de internet del sistema operativo [Android 2.2](https://es.wikipedia.org/wiki/Android) “Froyo”. Implementa [ECMAScript](https://es.wikipedia.org/wiki/ECMAScript" \o "ECMAScript) como especifica [ECMA-262](https://es.wikipedia.org/w/index.php?title=ECMA-262&action=edit&redlink=1) 5.ª edición y corre en [Windows XP](https://es.wikipedia.org/wiki/Windows_XP).
4. **V8 es fácil de construir y no viene con la máquina virtual de Java de sobrecarga de Mozilla independiente intérprete de Javascript. Por suerte, V8 se distribuye con código para la construcción de una consola.**
5. **A y B**
6. **A, B,C**

**Explicación**

V8 es un motor de código abierto para JavaScript creado por Google, siendo su programador jefe Lars Bak.3

Está escrito en C++ y es usado en Google Chrome. También el "V8 JavaScript" está integrado en el navegador de internet del sistema operativo Android 2.2 “Froyo”. Implementa ECMAScript como especifica ECMA-262 5.ª edición y corre en Windows XP, Vista, Mac OS X 10.5 (Leopard) y Linux en procesadores IA-32 y ARM.

V8 puede funcionar de manera individual (standalone) o incorporada a cualquier aplicación C++.

Hay varias ventajas:

1. No hay que hacer un cambio de chip en un mismo proyecto en la parte cliente y servidor. No hay lenguaje de cliente y lenguaje de servidor.
2. El intercambio de datos para aplicaciones Ajax tiene un formato ideal: JSON. Un formato nativo del lenguaje es lo mejor que puede pasar para simplificar el intercambio.
3. Su ligereza y su simplicidad en el desarrollo (sobretodo en comparación con Java, pero también con otros de Scripting como PHP, Ruby, Python).
4. Para todos los desarrolladores de javascript en la web se abre un campo inmenso en el cual expandirse.
5. **Verdadero o falso**

**Los objetos remotos especificados en la interfaz remota son accesible remotamente.**

1. Verdadero
2. **Falso**

**Explicacion**

Todo objeto remoto debe implementar esta interfaz. Solo los métodos especiados en la interfaz remota son accesibles remotamente.

1. **¿Qué es** UnicastRemoteObject**?**
2. La clase UnicastRemoteObject es la superclase común para implementaciones de servidor y proporciona el marco para apoyar una amplia gama de la semántica de referencia remotas . En concreto , las funciones necesarias para crear y objetos remotos de exportación (es decir, para que sean remotamente disponible) se proporcionan de manera abstracta por ServidorRemoto y concretamente por su subclase ( es) .
3. La clase UnicastRemoteObject implementa el comportamiento java.lang.Object para objetos remotos. RemoteObject proporciona la semántica remotas de objetos por métodos para hashCode , iguales, y toString ejecución.
4. Se utiliza para exportar un objeto remoto con JRMP y la obtención de un talón que se comunica al objeto remoto.
5. **A y B**
6. **Ninguna de las anteriores**

**Explicación**

public class **UnicastRemoteObject**

extends [RemoteServer](https://docs.oracle.com/javase/7/docs/api/java/rmi/server/RemoteServer.html)

Se utiliza para exportar un objeto remoto con JRMP y la obtención de un stub que se comunica al objeto remoto.