Medidas de Seguridad implementadas en Nepcom

Technical White paper.

Versión: Enero 2017

1) Medidas de seguridad generales

Servidores alojados en España. Aplicación del marco legal nacional y europeo

- Existe pleno cumplimiento de la normativa española y europea en materia de protección de datos.

- Los datos están alojados en Data Centers que cumplen altos estándares de seguridad. Los Data Centers están catalogados como Tier III. Un Data Center catalogado como Tier III garantiza una disponibilidad del 99,984%, esto supone menos de 1,6 horas de interrupción en el año. También cuenta con redundancia en sus infraestructuras y de varias fuentes alternativas de electricidad y refrigeración. Las operaciones de mantenimiento no requieren que el centro este offline en ningún momento.

- Todos los datos referentes a conversaciones y archivos adjuntos se eliminan de los servidores una vez transmitidos.

- Sólo se alojan y almacenan los datos estrictamente necesarios, como son los datos identificativos del perfil de los usuarios o la organización de las salas de chat.

- Se toman medidas de seguridad para garantizar la procedencia y validez de las peticiones que llegan al servidor.

Modo de conversaciones efímeras: Modo confidencial

- Especialmente indicado para tratar información sensibles o confidencial.

- Las conversaciones se eliminan de los dispositivos simultáneamente una vez abandonado el modo o minimizada la app. Para garantizar que la conversación sea completamente efímera, se han inhabilitado las capturas de pantalla[1], copiar y compartir mensajes recibidos y guardar imágenes o vídeos recibidos en la galería en este modo, siempre con el único objetivo de proteger la información.

Cifrado extremo a extremo

- Se cifran las conversaciones mediante protocolo OTR (explicado en detalle más adelante).

Otras medidas generales

- Uso de certificados para evitar suplantación de identidad y ataques del tipo Man-in-the-Middle.

[1]Se han inhabilitado las capturas de pantalla en los dispositivos con sistema operativo Android. En los dispositivos con sistema operativo iOS, dado que no resulta posible inhabilitar las capturas según los criterios de Apple, se ha establecido un mecanismo que notifica al otro interlocutor la realización de capturas de pantalla durante la conversación en modo confidencial.

- Base de datos interna de la app cifrada. Se cifra la base de datos mediante AES 256 bits.

- Mecanismos para evitar el escalado de privilegios.

- Seguridad perimetral implantada en los servidores.

- Las partes sensibles de la aplicación presentan código ofuscado.

2) Medidas específicas en los clientes

Los clientes tienen sus propias medidas de seguridad. En esta sección incluiremos los detalles técnicos del cifrado ya que es una funcionalidad de cifrado extremo a extremo que se realiza en los clientes.

Cifrado de conversaciones mediante protocolo OTR:

Nepcom utiliza el protocolo OTR (Off-the-Record Messaging) en su versión 3 para el cifrado extremo a extremo de las conversaciones. OTR permite conversaciones privadas en mensajería instantánea a través de redes de comunicación que proporcionen la entrega de los mensajes en orden. No tiene por qué ser completa, por ejemplo un usuario podría desconectarse en algún momento.

1.- Resumen general

El protocolo OTR está diseñado para ser utilizado en mensajería instantánea y proporciona un cifrado fuerte mediante una combinación de diferentes herramientas de cifrado que incluyen AES de clave simétrica, el protocolo de intercambio de claves mediante Diffie-Hellman y la función hash SHA1.

2.- Funcionalidades principales

Las funcionalidades principales son:

- Cifrado: sólo los legítimos destinatarios podrán leer tus mensajes.

- Autenticación: el destinatario es siempre quien dice ser.

- Ausencia de trazas/firmas: los mensajes que se envían no llevan trazas, ni firmas que pueda verificar un tercer actor. Durante una conversación, el actor secundario implicado puede estar seguro de que los mensajes que lee y recibe no vienen de un tercero, asegurando así su autenticidad y también que los mensajes no han sido modificados.

-Secreto: en el caso de pérdida de las claves privadas, ninguna conversación anterior quedaría comprometida.

3.- Solicitud de conversación OTR

Se puede informar a la otra parte de que se quiere iniciar una conversación vía protocolo OTR mediante el envío o de un mensaje especial denominado OTR Query Message. También se puede optar por incluir una etiqueta específica en un mensaje normal. En cualquiera de los dos casos existe la posibilidad de indicar la versión exacta del protocolo que implementa el emisor.

4.- Intercambio de claves de autenticación

La idea general es que un emisor A (Antonio) y un emisor B (Blanca) intercambian sus claves mediante el protocolo Diffie-Hellman (D-H) sin autenticación para configurar un canal cifrado y posteriormente hacer una autenticación mutua en el propio canal. (Todas las exponenciaciones de un número se realizan módulo un primo particular de 1536 bits, y g es un generador de ese grupo.).

Antonio inicia un intercambio de claves de autenticación con Blanca:

Antonio:

1. Selecciona un número aleatorio (r) de al menos 128 bits.
2. Selecciona un número aleatorio (x) de al menos 320 bits.
3. Envía a Blanca: $AES_r(g^x)$, $HASH(g^x)$

Blanca:

1. Selecciona un valor aleatorio (y) de al menos 320 bits.
2. Envía a Antonio: g^y

Antonio:

1. Verifica que el valor de Alicia (gy) es un valor legal (2 <= gx<= modulo-2)
2. Calcula s, siendo s = $(g^y)^x$
3. Calcula dos claves AES, c y c' y cuatro claves MAC m1, m1', m2 y m2' por hash de s en varias direcciones.
4. Selecciona un $keyid_B$, para su clave D-H g^x
5. Calcula $M_B = MAC_{m1}(g^x, g^y, pub_B, keyid_B)$
6. Calcula $X_B = pub_B, keyid_B, sig_B(M_B)$
7. Envía a Blanca r, $AES_c(X_B)$, $MAC_{m2}(AES_c(X_B))$

Blanca:

1. Usa r para descifrar el valor g^x enviado anteriormente
2. Verifica que HASH(g^x) coincide con el valor enviado anteriormente
3. Verifica que el valor de Antonio es un valor legal (2 <= gx<= modulo-2)
4. Calcula s = $(g^x)^y$ (es el mismo valor de s calculado por Antonio)
5. Calcula dos claves AES c y c' y cuatro claves MAC m1, m1', m2, m2' vía hash de s en varias direcciones (al igual que Antonio).
6. Usa m2 para verificar $MAC_{m2}(AES_c(X_B))$
7. Usa c para descifrar $AES_c(X_B)$ para obtener $X_B = pub_B, keyid_B, sig_B(M_B)$
8. Calcula $M_B = MAC_{m1}(g^x, g^y, pub_B, keyid_B)$
9. Usa pub_B para verificar $sig_B(M_B)$
10. Escoge $keyid_A$, su clave D-H g^y
11. Calcula $M_A = MAC_{m1'}(g^y, g^x, pub_A, keyid_A)$
12. Calcula $X_A = pub_A, keyid_A, sig_A(M_A)$
13. Envía a Antonio $AES_{c'}(X_A)$, $MAC_{m2'}(AES_{c'}(X_A))$

Antonio:

1. Usa m2' para verificar $MAC_{m2'}(AES_{c'}(X_A))$
2. Usa c' para descifrar $AES_{c'}(X_A)$ para obtener $X_A = pub_A, keyid_A, sig_A(M_A)$
3. Calcula $M_A = MAC_{m1'}(g^y, g^x, pub_A, keyid_A)$
4. Usa pub_A para verificar $sig_A(M_A)$

Si todas las verificaciones son exitosas, Antonio y Blanca saben cada uno las claves públicas Diffie-Hellman del contrario y comparten un valor s. Blanca está segura de que s es conocido por alguien con acceso a la clave privada correspondiente a pub_B, y de manera similar para Antonio.

5.- Intercambio de los datos

Se describe el método utilizado para proteger los datos intercambiados entre Antonio y Blanca (como se ha indicado anteriormente todas las exponenciaciones de un número se realizan modulo un primo particular de 1536 bits, y g es un generador de ese grupo.).

Supongamos que Blanca quiere enviarle un mensaje a Antonio:

Blanca:

1. Selecciona el valor más reciente de sus propias claves de cifrado D-H que Antonio ha reconocido haber recibido satisfactoriamente (al utilizarla en un intercambio de mensajes o en su defecto en el intercambio inicial de claves de autenticación). Sea key_A esa clave y $keyid_A$ su número de serie.

2.Si la clave anterior es la más reciente de Blanca, ella genera una nueva clave D-H (next_dh) para obtener el número de serie $keyid_A$+1.

3. Escoge la más moderna de las claves de cifrado de Antonio recibidas por ella (vía mensaje recibido o en su defecto en el intercambio inicial de claves de autenticación). Sea key_B esa clave y $keyid_B$ su número de serie.

4. Utiliza Diffie-Hellman para calcular un secreto compartido entre las dos claves key_A y key_B y genera la clave AES, ek, y la clave MAC de envío, mk

5. Recoge todas las claves anteriores MAC que se usaron en mensajes pasados y que nunca se volverán a usar (ya que sus claves D-H asociadas ya no son las más recientes) y las guarda en una lista, viejasclaves.

6. Selecciona un valor del contador, ctr para que la tupla (key_A, key_B, ctr) nunca sea la misma para más de un mensaje de datos enviado de Blanca a Antonio.

7. Calcula T_A = ($keyid_A$, $keyid_B$, next_dh, ctr, AES-$CTR_{ek,ctr}$(msg))

8. Envía a Antonio T_A, MAC_{mk}(T_A), viejasclaves

Antonio:

1. Usa Diffie-Hellman para calcular un secreto compartido de las dos claves etiquetas por $keyid_A$ y $keyid_B$, y genera la clave AES receptora, ek, y la clave MAC receptora, mk

2. Usa mk para verificarMAC_{mk}(T_A)

3. Usa ek y ctr para descifrar AES-$CTR_{ek,ctr}$(msg)

6.- Protocolo del socialista millonario (SMP)

Mientras se intercambian mensajes Antonio y Blanca, cualquiera de ellos puede ejecutar el protocolo del socialista millonario (en adelante SMP) para detectar ataques del tipo Man-in-the-Middle o de suplantación de identidad.

Supongamos que Blanca y Antonio tienen información secreta, x e y, y desean saber si x = y. El protocolo SMP permite que se pueda comparar x e y sin revelar más información adicional. En OTR los secretos contienen información sobre las claves públicas de autenticación a largo plazo de ambas partes, así como la información introducida por los propios usuarios. Si x es igual a y

significa que Blanca y Antonio introdujeron la misma información secreta y por lo tanto tienen que ser las mismas entidades que establecieron el secreto al principio de la comunicación. (Como se ha indicado anteriormente todas las exponenciaciones de un número se realizan modulo un primo particular de 1536 bits, y g1 es un generador de ese grupo.).

Asumiendo que Blanca empieza el intercambio:

Blanca:

1. Selecciona exponentes aleatorios a_2 y a_3
2. Envía a Antonio: $g_{2a} = g_1^{a_2}$ and $g_{3a} = g_1^{a_3}$

Antonio:

1. Escoge exponentes aleatorios b_2 y b_3
2. Calcula $g_{2b} = g_1^{b_2}$ y $g_{3b} = g_1^{b_3}$
3. Calcula $g_2 = g_{2a}^{b}$ y $g_3 = g_{3a}^{b_3}$
4. Escoge un exponente aleatorio r
5. Calcula $P_b = g_3^{r}$ y $Q_b = g_1^{r} g_2^{y}$
6. Envía a Blanca: g_{2b}, g_{3b}, P_b y Q_b

Blanca:

1. Calcula $g_2 = g_{2b}^{a_2}$ y $g_3 = g_{3b}^{a}$
2. Escoge un exponente aleatorio s
3. Calcula $P_a = g_3^{s}$ y $Q_a = g_1^{s} g_2^{x}$
4. Calcula $R_a = (Q_a / Q_b)^{a_3}$
5. Envía a Antonio: P_a, Q_a y R_a

Antonio:

1. Calcula $R_b = (Q_a / Q_b)^{b_3}$
2. Calcula$R_{ab} = R_a^{b_3}$
3. Comprueba si $R_{ab} == (P_a / P_b)$
4. Envía a Blanca R_b

Blanca:

1. Calcula $R_{ab} = R_b^{a_3}$
2. Comprueba si $R_{ab} == (P_a / P_b)$

Si todo se hace correctamente, entonces R_{ab} debe mantener el valor de (P_a / P_b) multiplicado por $(g_2{}^{a}{}_{3}{}^{b}{}_{3})^{(x-y)}$, lo que significa que la prueba final del protocolo solo tendrá éxito si x es igual a y. Además puesto que $g_2{}^{a}{}_{3}{}^{b}{}_{3}$ es un número aleatorio no conocido por ninguna parte, si x no es igual a y no se revela ninguna otra información.

3) Medidas en la comunicación entre clientes y servidores

La app de Nepcom se comunica con diferentes servidores propios y externos. Todas las comunicaciones son a través de protocolos seguros como HTTPS y SSL.

Conexión con servidor XMPP

Esta conexión se realiza mediante sockets SSL. SSL/TLS son unos protocolos criptográficos para realizar conexiones seguras entre un servidor y un cliente (en ambas direcciones). SSL es un protocolo a nivel de capa de transporte. Utiliza criptografía asimétrica.

Conexión con servidores push Google, Apple

Esta conexión se realiza mediante sockets SSL. Funciona de forma similar a lo indicado anteriormente

Conexión con servidor web

Esta conexión se realiza mediante HTTPS. HTTPS es la versión segura del protocolo HTTP. Utiliza SSL/TLS para establecer conexiones seguras en sitios web. Sirve para securizar HTTP. Es un protocolo a nivel de aplicación.