OpenID connect y la seguridad de la identidad digital

OpeniID Connect and the security of digital identity

Rubén De León-Peña[1] & Miguel Vargas-Lombardo[2]*

[1]Licenciatura en Redes Informáticas – Facultad de Ingeniería en Sistemas Computaciones – Universidad Tecnológica de Panamá, [2]Director del Grupo de Investigación en Salud Electrónica y Supercomputación-CIDITIC-Universidad Tecnológica de Panamá

Resumen Mediante soluciones de identidad federada los individuos pueden emplear la misma identificación personal (típicamente usuario y contraseña) para identificarse en redes de diferentes departamentos o incluso empresas. Vinculado al concepto, OpenID es un estándar de identificación digital descentralizado, con el que un usuario puede identificarse en una página web a través de una URL (o un XRI en la versión actual) y puede ser verificado por cualquier servidor que soporte el protocolo. Por su parte, el estándar también está enfocado hacia la privacidad. Aunque puede utilizarse para acceder a múltiples sitios, no se usa el mismo identificador para todos los sitios. Como resultado, muchas compañías han implementado OpenID Connect en todo el mundo, incluyendo Google, Microsoft, Deutsche Telekom, Salesforce, Ping Identity, así como otras empresas y organizaciones.

Palabras claves Autenticación, JSON Web Token, relaying party, OpenID Connect, OP Endpoints, Seguridad, Single Sign On, Token ID.

Abstract Through federated identity solutions individuals can use the same personal identification (typically user and password) to identify themselves in networks of different departments or even companies. Linked to the concept, OpenID is a decentralized digital identification standard, with which a user can be identified in a web page through a URL (or an XRI in the current version) and can be verified by any server that supports the protocol. For its part, the standard is also focused on privacy. Although it can be used to access multiple sites, the same identifier is not used for all sites. As a result, many companies have implemented OpenID Connect worldwide, including Google, Microsoft, Deutsche Telekom, Salesforce, Ping Identity, as well as other companies and organizations.

Keywords Authentication, JSON Web Token, relaying party, OpenID Connect, OP Endpoints, Security, Single Sign On, Token ID.

*** Corresponding author:** miguel.vargas@utp.ac.pa

1. Introducción

Con la proliferación de servicios en Internet, el problema de la multitud de sistemas de autenticación de usuarios y de acceso a los recursos y datos personales se hace cada vez más importante.

Esta revisión se organiza en 5 secciones: La sección 1 es la introducción, en la sección 2 se comentan los componentes del protocolo OpenID Connect. Luego en la sección 3 nos detallarán ataques y medidas preventivas; en la sección 4 se habla sobre casos de estudios, y en la sección 5 están las conclusiones.

2. OpenID Connect 1.0

El protocolo OpenID Connect representa un esfuerzo no sólo hacia la definición de protocolos abiertos, sino hacia la descentralización de la gestión de identidad y de datos personales.

2.1 Componentes básicos de OpenID Connect

2.1.1 Open Connect Core

Utiliza el protocolo OAuth para poder realizar la comunicación entre el usuario y un proveedor OpenID (OP). Para esta finalidad se emplea la estructura de datos Token ID, que está en formato JSON Web Token (JWT).

2.1.2 Open Connect Discovery

OpenID Connect Discovery es capaz de identificar el proveedor OpenID para el usuario final, el *relaying party* no necesita tener ningún conocimiento previo de su proveedor OpenID.

2.1.3 OpenID Connect Dynamic Client Registration

El *relaying party* debe poder registrarse con un proveedor OpenID, para que un cliente externo o usuario final utilice los servicios y recursos de OpenID Connect. OpenID Connect Dynamic Client Registration administra el proceso de los r*elaying parties* externos, registrando dinámicamente a sus clientes con el proveedor OpenID.

2.1.4 OpenID Connect Session Management

Este componente se utiliza para administrar la sesión de usuario final para OpenID Connect. OpenID Connect Session Management supervisa y actualiza el estado de inicio de sesión del usuario final con el proveedor OpenID, pero el *relaying party* puede cerrar la sesión de un usuario final quien es registrado desde el endpoint del proveedor OpenID [1].

2.2 OpenID Connect OP Endpoints

***Endpoint* de registro:** se utiliza para registrar un cliente con el OP, y de esta manera usar los servicios de OpenID Connect como autenticación.

***Endpoint* de autorización:** el usuario final tiene que autenticarse al OP por medio de un proceso de autenticación correspondiente y autoriza al cliente a acceder los recursos solicitados.

Endpoint* de *token: el cliente se comunica con el tokenEndp, para obtener el token ID y autenticar al usuario final. Esta comunicación se realiza directamente entre el cliente y el OP (sin la intervención del usuario final)

***Endpoint* de información de usuario (userinfoEndp):** retorna información acerca del usuario final autenticado como correo electrónico, dirección, teléfono, género, etc. a acceder a los recursos que usa el cliente como un *token* de acceso obtenido por medio de la autenticación OpenID Connect [2].

3. Seguridad sobre OpenID Conect

3.1 Espionaje y robo de *cookies y tokens*

Los *tokens* son almacenados en sesión local/ almacenamiento o en una *cookie* de cliente.

Cuando se crea una *cookie*, pueden ser interceptado en la red, o secuestrados por scripts XSS malicioso que son inyectados en cualquier página bajo el dominio del servicio.

Utilizan los llamados "*tokens* del portador" para comunicarse con sus actores. Estos *tokens* del portador son no codificables (ver figura 1).

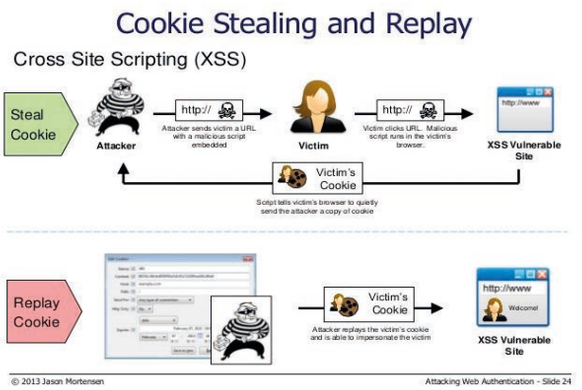

Figura 1. Robo de *cookies* [3].

3.1.1 Medidas preventivas

Para prevenir un ataque de robo de cookies, es posible enviar un parámetro escondido en las solicitudes de http.

En vez de *tokens* del portador, deberían ser usados los JSON web *tokens* más seguros.

Estos *tokens* pueden ser firmados y cifrados, que causa una solución que es protegida por capas múltiples de seguridad e intimidad.

3.2 Suplantación de *tokens*

Cuando se habla de una suplantación de token se refiere al enlace contextual, que significa que el servicio no proporciona un parámetro de estado durante la solicitud de autorización para establecer el estado entre la petición y la respuesta.

Este parámetro de estado es normalmente un valor que está enlazado a la sesión del navegador (ver figura 2).

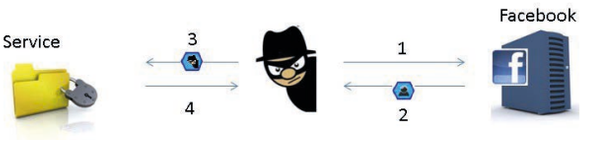

Figura 2. Suplantación de *tokens* [4].

En el cambio de sesión el atacante intercepta las credenciales SSO en el agente de usuario y completa el ataque incrustando las credenciales SSO interceptada en una construcción HTML (img, iframe).

Esto hace que el navegador envíe la credencial SSO interceptada al OP cuando la página *exploit* es vista por una víctima (ver figura 3).

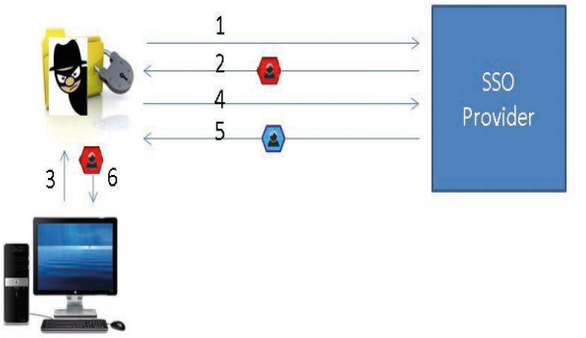

Figura 3. Suplantación de *tokens* [4].

3.2.1 Medidas preventivas

Estos ataques se pueden prevenir con el uso de nonces en la comunicación. En ese caso, el cliente detecta que el nonce devuelto por el OP no coincide con el enviado para iniciar sesión en el proveedor SSO.

3.3 Ataque de redirección abierta

Un ataque de redirección abierta es una aplicación que toma un parámetro (por ejemplo desde una URL) y redirige al usuario el valor asociado de ese parámetro sin ninguna validación.

Esta vulnerabilidad es utilizada en ataques de *phishing* para hacer que los usuarios visiten sitios maliciosos sin darse cuenta, como se muestra en la figura 4.

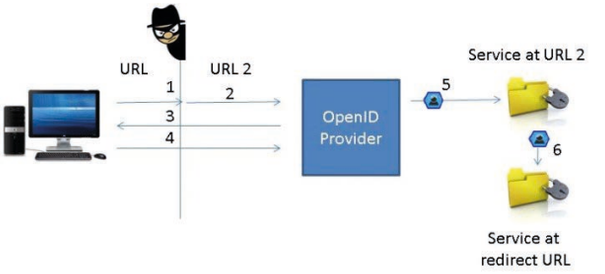

Figura 4. Ataque de redirección abierta [4].

3.3.1 Medidas preventivas

Comprobar si el código de acceso recibido proviene del dominio que emitió el *token* de acceso.

Utilizar una lista blanca que contenga todas las URLs de redirección permitida, para evitar que el atacante envíe un código de acceso a una dirección URL no válida; esto disminuye la probabilidad de conseguir *tokens* robados [4].

3.4 Suplantación de OP

La suplantación de OP utiliza un OP malicioso que crea un *token* de autenticación que contiene identificadores que el OP no puede controlar.

El atacante accede a la cuenta de una víctima desde el *relaying party* (Confiar en la Fiesta: Confiar en la sesión).

3.4.1Medidas preventivas

El *relaying party* deberá verificar la información descubierta. Para ello, debe verificar si el identificador introducido coincide con el identificador en el *token* de autenticación [5], ver figura 5.

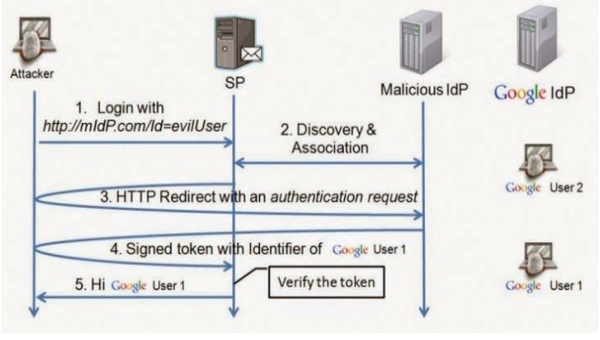

Figura 5. Suplantación de OP [5].

3.5 Ataque de confusión de llave

Explota una vulnerabilidad en la implementación de gestión de llaves del *relaying party*, dando por resultado el uso de una clave de confianza. El atacante actúa como un OP malicioso y utiliza la fase de asociación para establecer un secreto compartido con el *relaying party* de destino.

Posteriormente, el atacante confunde al *relaying party* de manera que pueda utilizar la clave compartida de otro OP honesto cuando en realidad, es el perteneciente del OP malicioso [6], por ejemplo (ver figura 6).

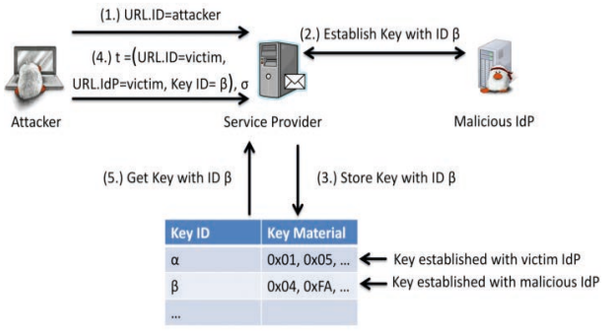

Figura 6. Ataque de confusión de llave [7].

3.6 Ataque de *endpoint* malicioso

Fase 1.1: inyección de *endpoints* malicioso: la intención del atacante en la primera fase es obligar a un cliente válido a usar el Discovery Service malicioso del atacante. Para ello, construye un vínculo malicioso y lo envía al usuario final.

Fase 1.2: registro dinámico: el cliente accede a regEndp para el registro dinámico. Se envía una solicitud de registro a https://honestOP.com/register y recibe un ID de cliente y un cliente secreto en la respuesta.

Fase 2: autenticación de usuario final y autorización: el cliente redirige al usuario final al authEndp, https://login.honestOP.com/, donde el usuario final tiene que autenticarse a sí mismo y autorizar al cliente.

Fase 3: el robo dependiendo del protocolo de flujo (código o implícito): los diferentes mensajes son enviados al atacante.

Este *endpoint* es un recurso protegido OAuth 2.0 que devuelve peticiones del usuario autenticado. El *token* de acceso obtenida se envía como un *token* de portador por el cliente. Por lo tanto, el atacante puede obtener acceso a un *token* de acceso válido [8], tal y como se muestra en la figura 7.

4. Casos de estudio con OpenID Connect

El modelo centrado en el usuario IdM es el más adecuado, según las necesidades de los beneficiarios del sistema de salud por las siguientes razones: la capacitación al usuario: los usuarios pueden tener control sobre el flujo de liberación del atributo en los *relaying parties.*

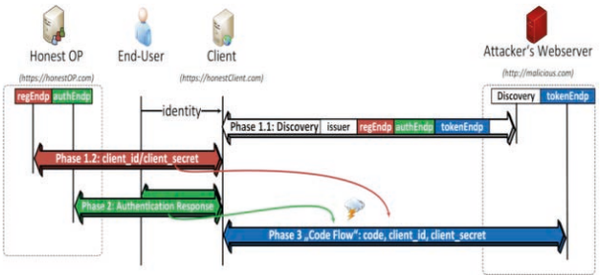

Figura 7. Ataque de *endpoint* malicioso [8].

Y por el otro lado, dar opciones al usuario: los usuarios pueden cambiar de OP sin preocuparse de perder el acceso a servicios de salud y finalmente, la privacidad: el seguimiento de la información de usuario se hace más difícil.

La *Smart Gateway* tiene un papel primordial en la implementación de una solución de IdM, más allá de actuar como un agregador de recursos de dispositivos y la disposición de dichos recursos a través de servicios web RESTful [9] – ver figura 8.

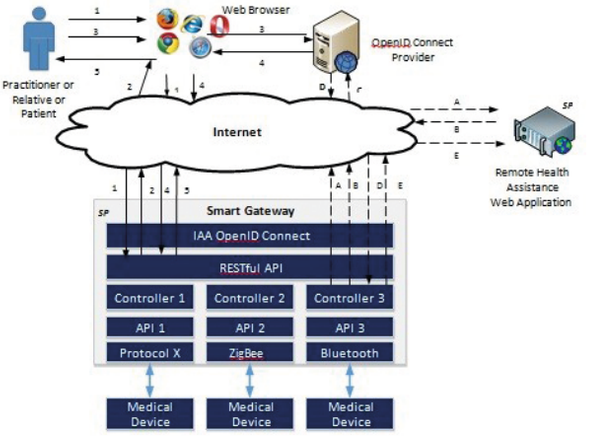

Figura 8. Uso de OpenID Connect como un CPS de asistencia médica remota [9].

4.1 Servicio de seguridad

El servicio de seguridad basado en la nube para los sistemas de diagnósticos de imágenes, la herramienta de código abierto "PYOIDC" es una implementación completa de las especificaciones de OpenID Connect escrito en Python. La herramienta de código abierto CherryWado simula un servidor WADO que también está escrito en Python. CherryWado usa un servidor Web Server Gateway Interface (WSGI) para implementar aplicaciones web de Python.

Las imágenes DICOM son manejadas por Python Imaging Library (PIL) que proporciona capacidades de procesamiento y gráficos de imagen poderosa. El servicio WADO no hace ninguna suposición sobre cómo/dónde se almacenan los ficheros DICOM [10]. Veamos el diagrama de la figura 9.

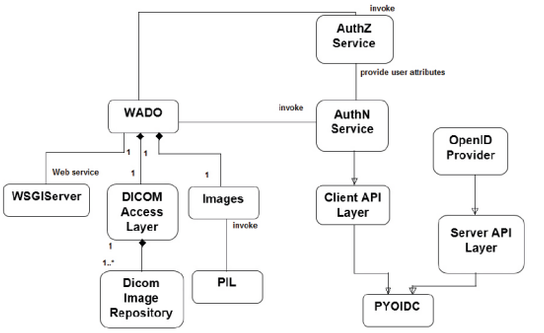

Figura 9. Diagrama de estructura del prototipo de implementación [10].

5. Conclusiones

OpenID Connect amplía el protocolo de autorización de OAuth 2.0 para usarlo como un protocolo de autenticación, lo que le permite realizar inicios de sesión únicos mediante OAuth. Presenta el concepto de un *token* ID, que es un *token* de seguridad que permite al cliente comprobar la identidad del usuario y obtener información del perfil básica sobre el usuario.

Puesto que amplía OAuth 2.0, también permite que las aplicaciones adquieran de forma segura *tokens* de acceso que se pueden usar para acceder a los recursos protegidos mediante un servidor de autorización. Existen diferentes problemas de seguridad hacia este protocolo, además, existen medidas preventivas las cuales han sido estudiadas para una mejor implementación.

Referencias

[1] A. Perera. (2014, July 3, 2014). An Introduction to Open ID Connect. Available: http://wso2.com/library/articles/2014/06/open-id-connect/

[2] V. Mladenov, C. Mainka, J. Krautwald, F. Feldmann, and J. Schwenk, "On the security of modern Single Sign-On Protocols: OpenID Connect 1.0," Cornell University Library, pp. 1-15, 2015.

[3] J. Mortensen. (2013, 23-10-2016). Web-based impersonation attacks. Available: http://www.slideshare.net/ConciseCourses/session-management-authentication

[4] Sign-On," Master, CCV, CCV & Radboud University Nijmegen, 2015.

[5] V. Mladenov and C. Mainka. (2014, 23-10-2016). Attacking SSO Part 1: ID Spoofing. Available: http://web-in-security.blogspot.com/2014/12/attacking-sso-part-1-id-spoofing.html

[6] C. Mainka and V. Mladenov, "Do not trust me: Using malicious IdPs for analyzing and attacking Single Sign-On," in 2016 IEEE European Symposium on Security and Privacy (EuroS&P), 2016, pp. 321-336.

[7] V. Mladenov and C. Mainka. (2015, 23-10-2016). Attacking SSO Part 2: Breaking OpenID in Drupal with Key Confusion. Available: http://web-in-security.blogspot.com/2015/01/attacking-sso-part-2-breaking-openid-in.html

[8] V. Mladenov and C. Mainka. (2015, 23-10-2016). Attacking OpenID Connect 1.0 - Malicious Endpoints Attack. Available: http://web-in-security.blogspot.com/2015/10/attacking-openid-connect-10-malicious.html

[9] M. C. Domenech, E. Comunello, and M. S. Wangham, "Identity management in e-Health: A case study of web of things application using OpenID connect," in e-Health Networking, Applications and Services (Healthcom), 2014 IEEE 16th International Conference pp. 219-224.W. Ma, K. Sartipi, H. Sharghigoorabi, D. Koff, and P. Bak, "OpenID Connect as a security service in cloud-based medical imaging systems," Journal of Medical Imaging, vol. 3, pp. 1-9, 2016.