SEGURIDAD PARA IOT, UNA SOLUCIÓN PARA LA GESTIÓN DE EVENTOS DE SEGURIDAD EN ARQUITECTURAS DE INTERNET DE LAS COSAS

Realizado por:

Nicolas Moreno Guataquira, Stefany Morón Castro y Andrés Felipe Vega Torres

PROGRAMA DE INGENIERÍA DE SISTEMAS
DEPARTAMENTO DE INGENIERÍA DE SISTEMAS
FACULTAD DE INGENIERÍAS

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

Bogotá, diciembre de 2017

|

ESCUELA COLOMBIANA DE INGENIERIA JULIO GARAVITO

RESUMEN

PROGRAMA INGENIERIA DE SISTEMAS

Realizado por Andrés Felipe Vega Torres, Nicolas Moreno Guataquira y Stefany Morón Castro

En este documento se puede evidenciar el trabajo de proyecto de grado titulado “Seguridad para IoT, una solución para la gestión de eventos de seguridad en arquitecturas IoT”, en donde se detalla el proceso para la elaboración de un dispositivo que permite tener unos servicios de seguridad para mitigar posibles ataques a los diferentes dispositivos IoT presentes en las arquitecturas de Smart Home. Por lo tanto se enfoca el tema de la seguridad de dispositivos IoT Smart Home de la siguiente forma, en la primera parte se abordó una manera general de tratar la seguridad de los dispositivos IoT desde una perspectiva no sesgada a la arquitectura anteriormente mencionada, se vieron diferentes formas de detectar los eventos de seguridad provenientes de los dispositivos IoT por medio de la plataforma SIEM, pasando por la caracterización de los dispositivos IoT, lo que los hace particulares y diferentes a otros dispositivos o activos de información, luego se definieron unas alertas y reglas de correlación para el caso general de los dispositivos IoT y así mismo se generaron respuestas a estas reglas de correlación. En una segunda parte del proyecto de investigación, se tiene una mirada más centralizada en los dispositivos IoT en las arquitecturas de Smart Home, comenzando por proponer herramientas de ciberdefensa como: Suricata, OpenVas y Kismet, que componen el dispositivo Centinela IoT, el cual brindará soluciones de seguridad a los dispositivos en mención. Para evidenciar lo anterior, se realizó un diagrama que explica la arquitectura de componentes del dispositivo Centinela IoT y su funcionamiento.Por último se establecieron reglas de correlación definidas como propias de los ataques posibles a la arquitectura Smart Home y se estructuraron las respuestas activas pertinentes a los ataques en mención, para así documentar el alcance del uso del dispositivo Centinela IoT en arquitecturas Smart Home.

PALABRAS CLAVES: Reglas de correlación; IoT; Smart Home; Centinela IoT; ciberdefensa; Kismet; Suricata; Openvas; respuestas activas; SIEM; Ataque; Evento.

|

TABLA DE CONTENIDO

TABLA DE ILUSTRACIONES

1. INTRODUCCIÓN

Se pretende proponer una solución para brindar seguridad a los dispositivos IoT en ambientes Smart Home, enfocando la investigación hacia la creación de un dispositivo llamado “Centinela IoT” que se encargará de extraer, generar y enviar eventos de seguridad de todos los dispositivos conectados en el ambiente Smart Home. Este dispositivo Centinela deberá tener 3 herramientas de ciberdefensa las cuales se investigarán para determinar cuáles son las mejores opciones para usarlas en el proyecto.

EL dispositivo Centinela deberá reportar los eventos de seguridad hacia una solución SIEM y en esta se deberá realizar la correlación de eventos, que para propósito de la investigación de tiene que cumplir la relación cruzada entre eventos de al menos dos de las herramientas contenidas en el dispositivo Centinela con el fin de predecir, detectar y contener ataques sobre dispositivos IoT Smart Home.

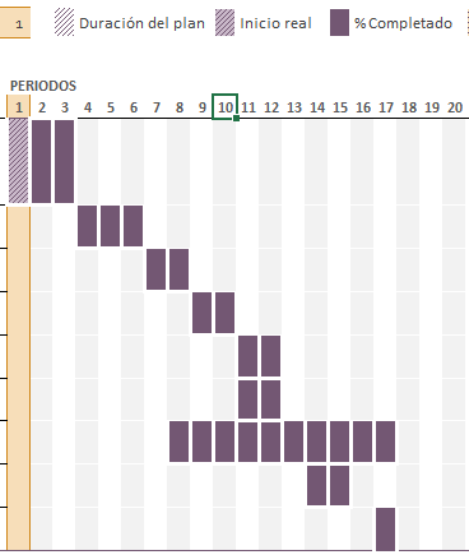
En el siguiente texto se abordará la investigación con la siguiente estructura, detección de eventos de seguridad, caracterización de activos IoT, alertas y reglas de correlación básicas para IoT, respuestas a incidentes de seguridad básicas, selección de herramientas para el Centinela IoT, arquitectura de la solución, directivas de correlación cruzada y su correspondiente respuesta.

CRONOGRAMA DEL PROYECTO

Cronograma de trabajo-IoT-PGR2

Seleccione un periodo para resaltarlo a la derecha. A continuación hay una leyenda que de Periodo resaltado: 1 Duración del plan Inicio real %Completado

ACTIVIDAD	INICIO DEL PLAN	DURACIÓN DEL PLAN	PORCENTAJE COMPLETADO
Investigación sobre herramientas de seguridad	1	3	100%
Configuracón de herramientas	4	3	100%
Implementación del OSSIEM	7	2	100%
Definición de la arquitectura del centinela	9	2	100%
Definición de reglas de correlación	11	2	100%
Implementación del centinela	11	2	100%
Desarrollo del segundo Articulo de soluciones SIEM para IoT.	8	10	100%
Preparación Vitrina academica(poster)	14	2	100%
Entrega del proyecto	17	1	100%



2. DETECCIÓN

La diversidad de dispositivos IoT ha crecido rápidamente y atrajo la atención del mundo de la industria y los círculos académicos. Internet of Things se considera una tecnología emergente con un gran potencial de desarrollo durante esta década [1]. IoT se considera como una parte de Internet del futuro y comprenderá miles de millones de 'cosas' comunicándose inteligentemente [2]. Estas tecnologías tienen características diferentes, como almacenamiento distribuido e incorporación de datos de tiempo y posición, y en muchos casos son responsables de realizar tareas críticas. Las soluciones IoT se componen de una red compleja de dispositivos inteligentes, sensores y conectividad a Internet, a través de los cuales se pueden recopilar, intercambiar y almacenar datos. Estos dispositivos pueden estar ubicados en vehículos, edificios, electrodomésticos o teléfonos celulares e incluyen software que les permite ser administrados [3]. Las tecnologías IoT se han utilizado ampliamente en la gestión de la cadena de suministro, fabricación, monitoreo ambiental, venta minorista, operaciones de estantería inteligente, industria de la salud, alimentación y restauración, industria logística, industria del turismo y viajes, servicios de biblioteca y muchas otras áreas [2]. Para el año 2016, Gartner estimó que había 6 mil millones de cosas conectadas y pronosticó que para 2020 21 mil millones de cosas estarían conectadas. [1] [4].

Ahora, los dispositivos en Internet of Things (IoT) generan, procesan e intercambian grandes cantidades de datos críticos para la seguridad, así como también información sensible a la privacidad, lo que los convierte en objetivos atractivos para los ataques [5]. Además de las superficies de ataque que pueden afectar los sistemas informáticos tradicionales y pueden aplicarse a entornos IoT, existen superficies de ataque particulares, con diferente complejidad y vulnerabilidades [6], en todas las capas abstractas de IoT: Sensores / actuadores (por ejemplo, ataques físicos y ingeniería inversa), red (por ejemplo, hombre en el medio y DoS), servicios / software (por ejemplo, ataques en tiempo de ejecución, ingeniería inversa y malware) y negocios (por ejemplo, ingeniería social, phishing y spoofing) [5]. Uno de los desafíos más importantes para convencer a los usuarios de que adopten tecnologías emergentes (es decir, servicios de IoT) es garantizar la seguridad y privacidad de los datos de los usuarios [7]. Por lo tanto, es importante trabajar en la definición de seguridad y privacidad desde las perspectivas social, legal y cultural, el mecanismo de confianza, la seguridad de la comunicación, la privacidad de los datos del usuario y la seguridad de los servicios y aplicaciones [2]. Es necesaria una fuerte protección de seguridad de IoT para evitar ataques y mal funcionamiento, ya que la seguridad de IoT desafía la resistencia a los ataques, la autenticación de datos, el control de acceso y la privacidad del cliente [19]. Por otro lado, en [8] el autor indica que "para las aplicaciones en IoT, la seguridad y la privacidad son dos desafíos importantes. Para integrar los dispositivos de la capa de detección como partes intrínsecas del IoT, la tecnología de seguridad efectiva es esencial para garantizar la seguridad y protección de la privacidad en diversas actividades tales

como actividades personales, procesos comerciales, transportes y protección de la información ".
En la actualidad existen diferentes investigaciones relacionadas con los sistemas de seguridad IoT, entre ellas están las relacionadas con la garantía de sensores y actuadores a nivel de hardware, que ofrecen un chipset de seguridad o un microcontrolador basado en ARM TrustZone [9], [10], en la privacidad de los datos utilizando cifrado, por ejemplo, cifrado PKI en dispositivos IoT m-Health [11] y CryptoSmart [26], y en autenticación de usuarios y componentes: métodos de autenticación incluidos en MQTT [3], ULMAP [12], tarjeta inteligente [13] y autenticación mutua de pares de red [14] Desde una perspectiva de diseño y funcional, [14] propone el uso de eventos de IoT para determinar las condiciones operativas, es decir, los puntos finales que brindan servicios críticos al usuario deben habilitarse con un umbral de advertencia que indica diferentes eventos y de esta forma ayudan a determinar el funcionamiento correcto de un servicio. [14] también sugiere que el punto final debe registrar estos eventos en una memoria persistente para garantizar que el usuario y el administrador puedan recuperar esta información más adelante y afirma que esta información debe tener un sello de tiempo. Como estos eventos tienen una relación con las características operativas, como la potencia, la temperatura, etc., estos también podrían usarse como una entrada para detectar o prevenir actividades maliciosas o riesgos de seguridad no evidentes [14]. Un ejemplo de esto es una tecnología de automatización del hogar que reacciona en función de la presencia física de un usuario, lo que permite a un atacante diferenciar fácilmente entre una casa muy poblada y una casa vacía.
Otros autores, como [15], proponen un sistema para analizar el evento IoT como un sistema "auto-similar" dentro del funcionamiento normal de sus objetos, y [16], quien propone un sistema de gestión de eventos de seguridad usando agregación de datos, firma digital de datos y enjambre algoritmo de enrutamiento.

Y la aproximación de SIEM para IoT se presenta en AMSEC [17] [18]. AMSEC es una propuesta que complementa el SIEM tradicional para ser aplicado a los escenarios de IoT. El Proyecto AMSEC tiene un enfoque matemático y tiene las siguientes características: i) repositorio de seguridad, ii) árbol de ataque sobre dispositivos, iii) gráfico de ataque de dispositivos, iv) análisis de tiempo real de ataque, v) modelo de análisis estocástico, vi) gráfico de ataque y uso de combinación de gráfico de dependencia, vii) cálculo de métricas de seguridad, viii) solución de soporte de decisión de seguridad basada en requisitos y objetivos de seguridad. El proyecto no presenta solo un modelo de captura de eventos de seguridad, sino que asume que el problema radica en la gestión de los eventos y no solo en cómo generar o transmitir estos eventos.
Otro trabajo relacionado se describe en [15], presenta un punto de vista diferente sobre SIEM para dispositivos IoT, en este caso los autores buscaron detectar eventos de seguridad y cómo esos eventos fueron parte de un incidente a través del estudio de una arquitectura IoT como un dispositivo de red nativo. Esta red se representó como un gráfico con nodos (dispositivos) y enlaces (interconexión entre

dispositivos) y el análisis de seguridad consiste en la medida de anomalías gráficas que utilizan geometría fractal.

Como se ha visto anteriormente, todavía hay muchos desafíos por resolver en la seguridad de Internet de las cosas y una de las alternativas para avanzar en la seguridad del servicio IoT podría ser utilizar de una manera más integral los eventos de seguridad generados por los componentes de un Ecosistema de la IoT. Los eventos de seguridad podrían procesarse y correlacionarse para identificar actividades sospechosas sobre las diferentes superficies de ataque de IoT. Además, se deben aplicar esfuerzos no solo para identificar incidentes sino también en la respuesta a través de acciones defensivas que pueden establecerse para disminuir el impacto.

|

3. CARACTERIZACIÓN DE ACTIVOS

En esta sección se va a trabajar los aspectos que hacen a los dispositivos IoT únicos. Los temas se van a abordar de la siguiente forma: amenazas de los dispositivos IoT, vulnerabilidades de los dispositivos IoT, los distintos eventos que un dispositivo IoT puede generar y la información que puede manejar un dispositivo IoT.

Uno de los aspectos diferenciadores de un dispositivo IoT es que tienen como finalidad ser usados en momentos que se requiera tener conectividad que genere información en tiempo real, ya sea en sensores, maquinas fijas o móviles, es importante tener esto claro para poder categorizar sus activos de información ya que son elementos con capacidades limitadas lo cual brinda nuevas formas de ser vulnerados y generan activos propios mucho más reducidos y específicos.

Al hablar de las amenazas que pueden explotar las vulnerabilidades presentes en los dispositivos IoT, cabe decir que las amenazas contempladas se basan en las amenazas vistas anteriormente en la sección de marco teórico.

Las amenazas en los dispositivos IoT se evidencian desde tres vistas, la primer comprende las amenazas directas hacia el dispositivo como un hardware, la segunda es hacia las comunicaciones entre diferentes dispositivos de la arquitectura IoT (dispositivos IoT, servidores de análisis y recolección de datos, dispositivos de red, etc.), finalmente, desde la perspectiva de ataque a los servidores que controlan las tareas de los dispositivos IoT.

Desde la perspectiva de amenazas sobre el dispositivo como un hardware podemos encontrar que estas amenazas se caracterizan por explotar las vulnerabilidades en cuanto a la carencia o poca seguridad física del dispositivo, observando que el dispositivo IoT se caracteriza por tener una gran facilidad de acceso tácito a el mismo, ejemplos como tener acceso a un celular personal de una persona o un sensor que mide la humedad del aire, permitiendo manipular estos dispositivos hacia el atacante tener acceso libre a controlar todo el hardware que compone estos dispositivos CiberFisicos.

Observando el dispositivo IoT desde la perspectiva de la comunicación entre dispositivos de la misma arquitectura IoT se encuentran una gran variedad de vulnerabilidades, unas que afectan netamente al ámbito de red y otras relacionadas al funcionamiento del dispositivo. Por parte de las vulnerabilidades expuestas por medio de la red, encontramos que existen amenazas que explotan vulnerabilidades relacionadas con interfaces del dispositivo, puertos abiertos, falta de cifrado en la capa de trasporte e incluso configuraciones erróneas de los dispositivos que habilitan la capacidad de un atacante a realizar un ataque DOS (Denial of Service). También se encuentran amenazas que explotan vulnerabilidades respecto al funcionamiento del dispositivo, métodos de autenticación o autorización pobres o

falta de control en los inputs de las aplicaciones de los dispositivos se pueden evidenciar dentro de estas características.

En la perspectiva de la comunicación con los entes de control o reporte de los dispositivos IoT se encuentran amenazas que explotan vulnerabilidades netamente categorizadas por OWASP como interfaces inseguras con la nube [19], pero hace falta mencionar que dentro de las amenazas se pueden encontrar carencia de seguridad en la capa de trasporte y de aplicación por parte de estos servidores que representan la funcionalidad de recolección de datos de los dispositivos IoT o que simplemente ejecutan alguna labor de generar ordenes sobre estos dispositivo IoT.

Las principales vulnerabilidades de IoT basados como anteriormente se mencionó en el estudio realizado por OWASP, son las siguientes:

1. Username Enumeration: El atacante puede a través de esta vulnerabilidad puede recopilar un conjunto de nombres de usuarios validos interactuando con el mecanismo de autenticación.

2. Weak Passwords: Esta vulnerabilidad es una de las más comunes, donde se le asigna a determinado sistema un mecanismo de autenticación donde las contraseñas no cumplen con los requisitos de seguridad, y utilizan contraseñas como por ejemplo: 123, 1234567890, 777777, Mynoob, google, entre muchas otras y hasta a veces contraseñas predeterminadas del programa que con un simple ataque de diccionario sería muy fácil obtenerlas.

3. Account Lockout: En esta vulnerabilidad se puede encontrar un gran fallo de seguridad en el mecanismo de autenticación ya que se puede permitir la continuidad de intentos de iniciar sesión sin un límite de fallos.

4. Unencrypted Services: Esta vulnerabilidad se da cuando los servicios de red no están debidamente cifrados y esto permite que cualquier atacante que quiera pueda intervenir y escuchar lo que esté pasando por la red.

5. Two-factor Authentication: Se da cuando solo se cuenta con un mecanismo de autenticación por ejemplo con solo la contraseña. Dado que sola contraseña no es suficientemente seguro, un debido manejo de este mecanismo seria utilizar un Segundo, como por ejemplo un token, sistemas Biométricos, código de verificación entre otros.

6. Poorly Implemented Encryption: En esta vulnerabilidad se cuenta con un Sistema de cifrado, donde no está configurado correctamente o no se lleva un control de actualizaciones adecuado, haciéndolo débil y que será más fácil vulnerarlo.

7. Update Sent Without Encryption: Se da cuando se transfiere través de la red actualizaciones sin usar certificados ya sea TSL o SSL los cuales permiten y garantizan el intercambio de datos en un entorno seguro o sin cifrar el propio archivo de actualización.

8. Update Location Writable: La ubicación de almacenamiento para los archivos de actualización es compatible con el mundo potencialmente permitiendo que el firmware se modifique y distribuya a todos los usuarios.

9. Denial of Service: Esta vulnerabilidad provoca la pérdida de la conectividad o disponibilidad de la red por consumo masivo del ancho de banda de la red o sobrecarga de los recursos del dispositivo.

10. Removal of Storage Media: En esta vulnerabilidad se expone la importancia de asegurar no solo la parte del software, sino también la parte física del dispositivo, ya que existe la posibilidad que el atacante robe el soporte de almacenamiento del dispositivo.

11. No Manuel Update Mechanism: No tiene capacidad para forzar manualmente una comprobación de actualización para el dispositivo.

12. Missing Update Mechanism: En esta vulnerabilidad se muestra la importancia de las actualizaciones que ya si no se tiene un mecanismo de actualización funcionando , se pude dejar expuesto el dispositivo a posibles ataques, debemos tener en cuenta que cuando se hacen las actualizaciones se están realizando muchas tareas como: agregar funciones nuevas , eliminar funciones des-actualizadas , actualizar controladores, proporcionar correcciones de errores y, lo que es aún más importante , reparan vulnerabilidades detectadas.

13. Firmware Version Display and/or Last Update Date: Cuando no se puede ver las versiones actuales del firmware ni se muestra la última actualización, no se puede llevar un control versiones y esto es importante ya que si por ejemplo entra alguien que no tenga conocimiento de las actualizaciones previas no va saber si ese firmware se debe actualizara o no, y va a quedar expuesto a vulnerabilidades que la actualización debería reparar.

14. Firmware and storage extraction: Esta puede ser uno de los componentes más sensibles ya que guarda información como código fuente, contraseñas predefinidas, servicios en ejecución, claves de ssh, entre muchas otras.

15. Manipulating the code execution flow of the device: Con la ayuda de un adaptador JTAG y gdb se puede modificar la ejecución del firmware en el dispositivo y evitar casi todos los controles de seguridad basados en software.

Los ataques de canal lateral también pueden modificar el flujo de ejecución o pueden usarse para extraer información importante del dispositivo.

16. Obtaining console access: Esta vulnerabilidad permite que el atacante al conectarse desde una interfaz sere, obtenga acceso completo a la consola de un dispositivo. Para enviar este ataque por lo general las medidas de seguridad incluyen cargadores de arranque que no permiten que el atacante entre de modo de usuario único

Entrando al campo de los diferentes eventos generados por los dispositivos, se tienen los siguientes respecto a un estudio realizado por OWASP[20], en este estudio se plantea una categorización de los tipos de eventos que un dispositivo IoT puede llegar a generar, en donde los eventos se plantean en forma de excepciones que será objeto de estudio como eventos. Aquella categorización propuesta por OWASP es la siguiente:

1. Request Exceptions: En esta categoría se manejan aquellos eventos relacionados a solicitudes al dispositivo se no sean normales dentro del contexto del funcionamiento del mismo dispositivo, se encuentran referencias respecto a métodos HTTP no soportados o el número de parámetros que no es normal para solicitudes de recursos HTTP.

2. Authentication Exception: Dentro de esta categoría se agrupan eventos relacionados a la autenticación y sin importar su método de autenticación, como ejemplo podemos colocar el número excesivo de intentos para autenticarse a un recurso del dispositivo o aplicación, también se evidencias métodos como la geolocalización.

3. Session Exceptions: Dentro del funcionamiento de los dispositivos se encuentra la particularidad que para no realizar operaciones de autenticación se utiliza un modelo basado en sesiones, estas sesiones tienen la particularidad de poder garantizar mediante tokens la autenticación de un dispositivo frente al consumo de un recurso [20], para el caso de los dispositivos IoT, esta categorización muestra cuando puede haber una alteración dentro del token de sesión o session ID.

4. Access Control Exceptions: En esta categoría se encuentran aquellos eventos en donde se realizan tácitamente hablando intentos de ataques relacionados con la manipulación de parámetros de métodos GET y POST, así como el intento a acceder a referencias de objetos directamente.

5. Ecosystem Merbership Exceptions: Para esta categoría se encuentran aquellos eventos relacionados a la particularidad que tenga la arquitectura de los dispositivos, que generan ciertos eventos relacionados a si un dispositivo pertenece a la arquitectura de los dispositivos en cuestión.

6. Device Access Events: En esta categoría se encuentran los eventos generados al ocasionarse un acceso al dispositivo físico como lo son la perpetración de quitar algún elemento protector del dispositivo o la manipulación del hardware incorporando nuevos elementos.

7. Administrative Mode Events: Dentro de esta categoría se hace referencia al uso de privilegios de administrador para realizar alguna acción, pero lo importante no es la acción en concreto, es el hecho de elevar permisos de administrador, incluso si se hace con credenciales de administrador por defecto que se hayan dejado accidentalmente en la fase de desarrollo, pruebas o producción.

8. Input Exceptions: Este tipo de eventos hace alusión al erróneo intento de ingresar datos a aplicaciones o dispositivo en sí, incluso se llegan a contemplar errores de codificación en los diferentes inputs que puede tener una aplicación.

9. Command Injection Exceptions: Esta categoría es muy usual en cualquier tipo de activo de información, sin embargo, se considera que es de especial cuidado ya que representa una gran facilidad para ser explotado [23]. Para el caso de los dispositivos IoT, cualquier tipo de inyección de código puede ser perjudicial, pero debido a que se presentan un numero de dispositivos que no tienen interfaz de input para realizar alguna inyección SQL, resulta de especial interés los eventos generados por inyección de comandos, esto resulta ya que todos trabajan sobre un sistema operativo y este puede ser una superficie de ataque que resulta que todos los dispositivos tienen.

10. Reputation Exceptions: Dentro de esta categoría se encuentran eventos relacionados netamente con la ubicación geoespacial del dispositivo, incluso valores obtenidos por medidores como el giroscopio o acelerómetro resultan ser validos como características que generan eventos dentro de esta categoría, para sintetizar, la característica que agrupa esta clase son los valores anómalos de ubicación y posición del dispositivo.

11. Honey Trap Exception: Esta categoría es una trampa para el atacante ya que su funcionalidad es poner datos que sean tentadores para el atacante y este quiera hacer algún tipo de ataque ya sea modificarlos o robarlos, sin que él se percate que estos datos son falsos. Algunos posibles escenarios de ataque pueden ser: Un formulario, Por medio de una URL, valores de Cookie o encabezados de HTTP.

En cuanto a la información que puede tratar, generar o recolectar un dispositivo IoT depende netamente de su fin, se encuentra ejemplos de aplicaciones para la

industria, así como casos de uso en escenarios de personas usando este tipo de dispositivos.

El primero escenario referente a la información que maneja un dispositivo IoT se evidencia en la vida diaria de las personas, la información personal. Dentro de la información personal se encuentran 3 categorías de tipo de información que son aplicables para los dispositivos IoT: Información personal identificable, información personal de salud

En cuanto a la información personal identificable o PII por sus siglas en inglés (Personally Identifiable Information), que es toda aquella información que puede ser usada para identificar, contactar o localizar cierta persona o que esta información puede servir de ayuda para relacionarla con otra para identificar a la persona en cuestión; pero en el caso en particular de los dispositivos IoT, esta información se ejemplifica más específicamente en aquellos dispositivos que están en contante uso por personas, los celulares o smartwatches aquí se les ve como un ejemplo tácito, donde estos dispositivos manejan información en cuanto a la ubicación de personas, credenciales frente a distintos portales web o incluso datos personales tales como mensajes.

En cuanto a la información personal de salud o PHI por sus siglas en inglés (Personal Health Information), que es cualquier información que detalle sobre el estado pasado, actual o futuro de la condición mental o física de un individuo [32]; para el caso de los dispositivos IoT, este tipo de información se evidencia en aquellos dispositivos que encajen en la definición antes descrita sobre la información que ellos manejan, un ejemplo claro son aquellos dispositivos que permiten realizar seguimientos médicos o que muestran de manera clara valores de condición física de un individuo tales como dispositivos que monitorean el pulso de una persona, entre otros.

Por otra parte, otro escenario que permite evidenciar la información que puede tratar los dispositivos yace en el entorno de la industria, más precisamente, en sistemas SCADA. La información tratada en este escenario por parte de los dispositivos esta contextualizada por el hecho de que estos dispositivos están en un lugar específico dentro de una empresa, pero al mencionar especifico se detalla que no hace referencia a un lugar en particular, sino a la particularidad de su rol dentro de una línea de producción o cualquier área de una empresa o industria en general. La información además de ser acotada por el rol que cumple el dispositivo, también está acotada por aquellos aspectos de la información en particular, esto se entiende como la información específica dentro de una industria, que pueden ser datos de un proceso de manufactura o hasta información de la distribución de cierto producto dentro de una red especifica de entrega[21].

Basados en la información recopilada en esta sección hemos propuesto una posible relación entre superficies de ataque, vulnerabilidades y mensajes de eventos

explicados anteriormente con el fin de entender qué clase de eventos podemos correlacionar para detectar posibles ataques a el dispositivo IoT, también saber qué tipo de vulnerabilidad explota este ataque y desde donde se podría estar generando, y poder generar la respuesta ideal al posible ataque. Figure 1.

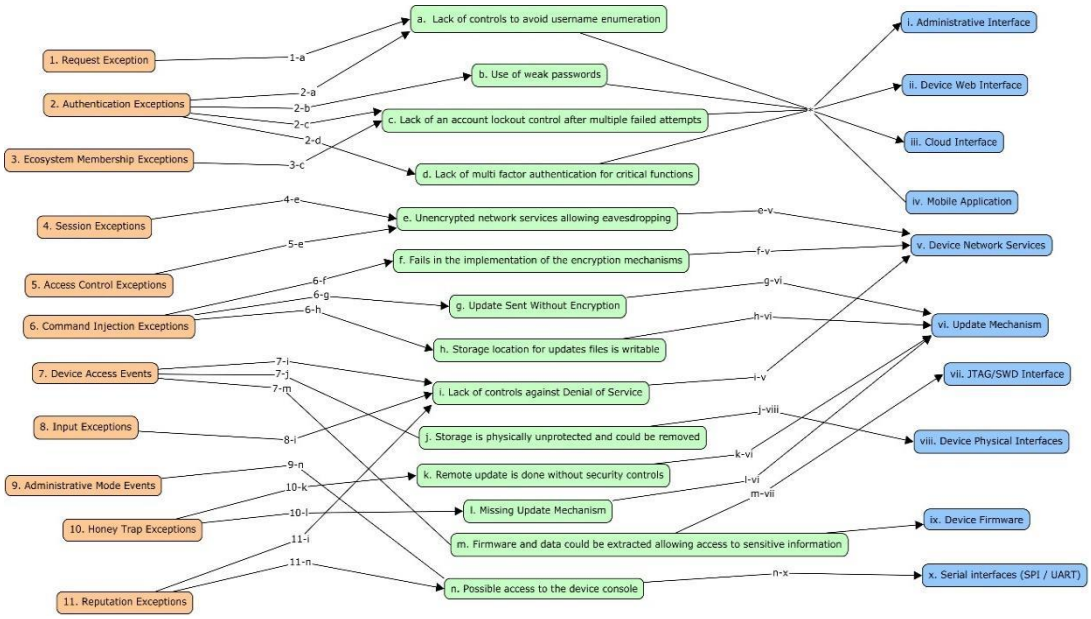


Ilustración 1Grafo de relación entre mensajes de eventos vs vulnerabilidades vs superficies de ataque.

4. ALERTAS Y REGLAS

Para esta sección se pretende abordar el tema de alertas y reglas de correlación que se pueden generar a partir de los eventos que los dispositivos IoT producen; para tal fin, va a estudiar tres alertas que son de especial interés para el estudio y como característica tienen que son simples, que son reglas que se originan a partir de eventos, pero al final suponen una ayuda para el administrador de la arquitectura IoT.

Geofencing

La primera regla de correlación o alerta se da en el escenario donde se tiene un dispositivo IoT que cuenta con la propiedad de poder moverse dentro de una zona delimitada a la cual se le llama zona segura, este dispositivo al tener la capacidad de poder moverse, un usuario malintencionado o atacante toma posesión del dispositivo para quedárselo, lo que se describe como el robo de un dispositivo IoT. Dentro de este escenario se tiene que se genera un tipo de evento por parte del dispositivo, y este es el evento relacionado con la ubicación del mismo, pero para generar la alerta es cuando el motor de correlación determina si esta fuera de la zona segura. (Ilustración 1, Escenario Geofencing)

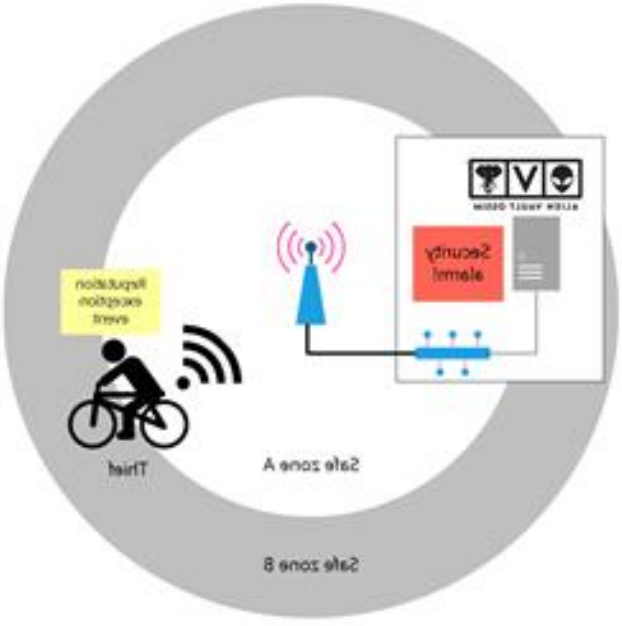


Ilustración 2, Escenario Geofencing

Para la anterior regla de correlación se debe tomar en cuenta que se está explotando la vulnerabilidad de denegación de servicio en contra del dispositivo IoT, además que tomar los eventos de ubicación que caben dentro de la categoría de Reputation Exceptions.

Ataque de fuerza bruta

La segunda regla de correlación, cabe en el escenario de que un atacante intenta realizar un ataque de fuerza bruta, el cual consiste en que decide probar con múltiples combinaciones de usuarios y de contraseñas hasta encontrar un usuario valido y su respectiva contraseña, lo primero que nos puede alertar de esto, es mediante un evento que aparece constantemente sobre usuarios inválidos durante un lapso de tiempo muy pequeño, esto estaría plasmado en el evento de authentication exception y session exception . Luego que el atacante encuentra un usuario valido, este procede a probar las múltiples contraseñas, para esto una autenticación fallida de claves erróneas en su respectivo lapso de tiempo nos terminara de asegurar que está sucediendo un ataque de fuerza bruta sobre un determinado dispositivo IoT. Es importante evitar este tipo de ataque porque una vez el atacante entre a el sistema como un usuario valido podría explotar nuevas vulnerabilidades ya estaría como un usuario de confianza dentro de nuestro sistema. (Ilustración 2, Escenario ataque fuerza bruta)

Cabe resaltar que esto estaría explotando diferentes vulnerabilidades como username enumeration, weak passwords y podría generar un account lockout.

Ilustración 3, Escenario ataque fuerza bruta

|

Inyección de Comandos

La tercera regla de correlación, se da en el escenario donde el atacante desea acceder o modificar datos por medio de un ataque de command injection, como lo dice su nombre es cuando se hace inyección de comandos que es interpretado o ejecutado por la aplicación del dispositivo IoT, lo que nos puede alertar de este tipo de ataque es el reconocimiento de caracteres no usuales(como ejemplo: código SQL) en los datos que admite el dispositivo IoT, y esto es debido a que no se tiene una apropiada validación de las entradas y salidas de los datos y por esta razón es vulnerable. El evento que se genera es Command injection Exception ya que se estaría dejando modificar argumentos ya sea para obtener o modificar los dados. (Ilustración 3, Escenario inyección de comandos)

Este ataque explota la vulnerabilidad de Denial of Service, ya que el servicio puede ser atacando de manera que genere denegación de servicio y dejar los recursos inaccesibles.

La regla se definió en OSSIM de la siguiente forma:

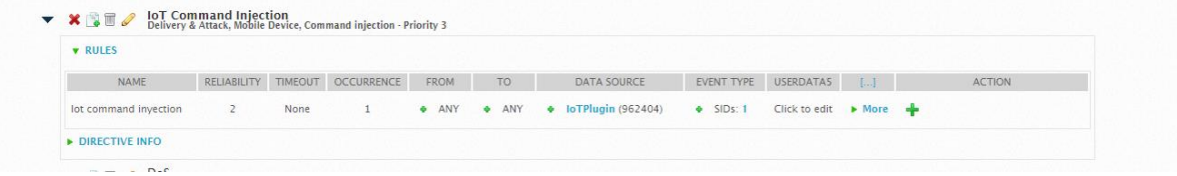


Ilustración 4 Regla de inyección de comandos

Donde se define en las veces que puede ocurrir el evento en este caso es una vez ya que sería extraño que un usuario se identificara con scripts o comandos SQL.

Ilustración 5, Escenario inyección de comandos

5. RESPUESTAS A INCIDENTES DE SEGURIDAD

Para esta sección de respuestas a incidentes las cuales son las acciones de seguridad defensiva que se deben tomar para neutralizar el ataque al dispositivo IoT, se mostrará para cada regla definida en la sección anterior, que respuesta se debe dar para cada una de ellas:

Geofencing

En referencia al **primer escenario**, donde se tiene que se intenta realizar un robo del dispositivo IoT, mediante la correlación de eventos propios de este escenario tenemos como respuesta diferentes posibles respuestas, esto se debe porque existen diferentes dispositivos IoT. Para este caso en particular se tomaron en cuenta dos opciones de posibles respuestas, la primera respuesta se toma como el apagar el dispositivo IoT, esto con el fin de evitar que el ya no se pueda hacer uso del propio dispositivo; la segunda respuesta está contemplada con el hecho de realizar un borrado de la información que posee el dispositivo, incluyendo hacer un borrado a nivel profundo, así ninguna información de configuración o de funcionamiento del mismo sea comprometida. Aquí cabe recordar que sea cual sea la respuesta que se elija para un dispositivo en particular, siempre de debe tomar en cuenta generar una alarma al administrador del dispositivo que permita al este actor estar informado de la situación que está ocurriendo.

|

Ataque de fuerza bruta

Para el **segundo escenario** de ataque de fuerza bruta al dispositivo IoT, el cual se puede controlar a través de diferentes respuestas, para la primera respuesta y más ideal está en habilitar la generación de un mensaje captcha en cada intento de autenticación del usuario luego que se determine que el ataque de fuerza bruta es una realidad, y como segunda posibilidad tenemos el bloqueo de la IP mediante el cual se está generando tal magnitud de intentos fallidos para evitar que siga generándolo, esta medida no es tan eficiente ya que puede evitarse al atacante cambiar de dirección IP una vez que esta sea bloqueada.

Para cualquiera de estas dos respuestas planteadas para el segundo escenario estarán acompañada de un correo electrónico al titular del dispositivo IoT mediante el cual se le hará previo aviso sobre el posible ataque, la información del dispositivo, de donde proviene el ataque y la respectiva medida que se ha implementado desde el motor de correlación de eventos.

Inyección de Comandos

Para el **tercer escenario** recordando que es un ataque de command injection, la respuesta y seguridad defensiva que se implementará será, que apenas se detecte que se está generando este ataque, ya sea porque están tratando de acceder a datos del dispositivo IoT por medio de comandos que generalmente un usuario no digitaria como lo son comandos de SQL, ya sea para modificarlos, robarlos o incluso eliminarlos, con respuesta inmediatamente se enviará un correo al administrador del dispositivo informándolo que el dispositivos está siendo atacado y mostrándole el tipo de ataque, hora y fecha en que ocurre.

Pero se debe tener en cuenta que se tienen varias posibilidades de respuestas como apagar el dispositivo, bloquearlo, activar foto instantánea y que esta se suba a la nube, entre muchas más, pensando cual es la respuesta adecuada para cada dispositivo IoT.

|

Prueba escenario tres:
Como prueba de este escenario se utilizó un dispositivos móvil en el cual por medio de un aplicación Android, se realizó el proceso de autenticación de usuario, en este caso el atacante inyecta código SQL y Scripts con el fin de falsificar la autenticación.

Realizar un ataque de Command injection a una app móvil(IoT)

Ilustración 6 Escenario ataque de invección de comandos

Cuando se detecta el ataque se envía el evento al OSSIM:

```
May 12 16:33:11 10.2.65.8  IOT: |<script> jdjdjd|CommandInjectionException|AE3|192.168.1.122|68:C4:4D:A4:80:9D|null|null|Fri May 12 11:33:11 GMT-05:00 2017
```

```
IOT: |<script> jdjdjd|CommandInjectionException|AE3|192.168.1.122|68:C4:4D:A4:80:9D|
```

Al momento en que recibe el evento genera una acción:

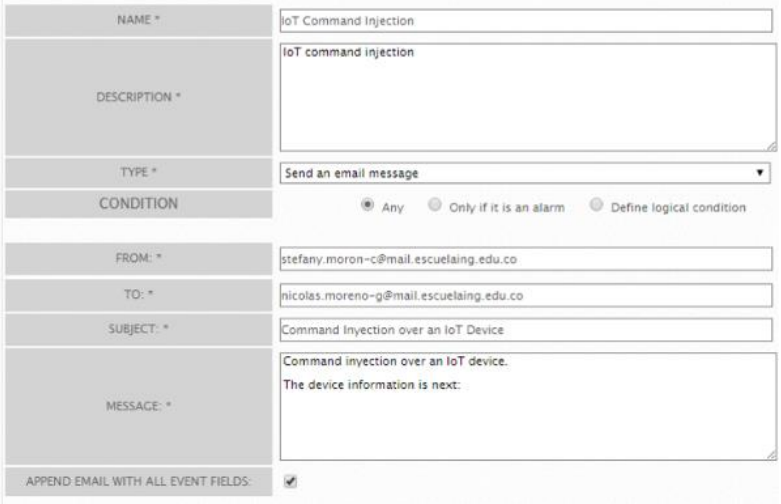


Ilustración 7 Respuesta a ataque inyección de comandos

El cual como respuesta le envía un correo al dueño o al administrador informándolo de que hay un posible robo de identidad y reaccionar frente a este ya sea, cambiando la contraseña.

|

6. SELECCIÓN DE HERRAMIENTAS CENTINELA IOT

Para la selección de las herramientas que contiene el dispositivo Centinela IoT se basó en aquellas herramientas que son utilizadas por la plataforma de OSSIM, para ello se hizo una evaluación en donde se priorizo aquellas herramientas que pueden ser de mayor utilidad para observar e identificar los ataques sobre los dispositivos IoT, para ello cabe resaltar que los dispositivos IoT normalmente se catalogan como dispositivos de caja negra, es decir, que no se puede manipular y que en su gran mayoría carecen de una consola de administración.

Por lo anterior, en el anexo 1 (tabla de herramientas de OSSIM) evidenciamos que las herramientas de Kismet y Suricata permiten analizar el tráfico, tanto de red como WiFi, y Openvas permite analizar las vulnerabilidades de los dispositivos, por lo tanto, se tiene la posibilidad de enfrentar los ataques desde el análisis de vulnerabilidades, la detección de tráfico malicioso y la identificación de ataques WiFi.

6.1 KISMET

Kismet es un detector de red y a su vez un detector de intrusos (IDS) para redes inalámbricas (cualquier LAN inalámbricas 802.11), puede detectar punto de acceso inalámbrico, clientes inalámbricos y asociarlos entre sí, la tarjeta del dispositivo donde opere debe estar en modo monitor. Se basa en Alertas predeterminadas para monitorear el tráfico que va pasando por una red WIFI y alerta de este modo para reportar que se está generando algún tipo de ataque, por algún punto de acceso detectado.

Instalación de Kismet

Para la instalar kismet en Linux solo es escribir los siguientes comandos en la terminal:

- **Sudo apt-get update**
- **Sudo apt-get Install kismet**

Como el IDS Kismet fue instalado en una Raspberry Pi 3, es importarte saber que la tarjeta de red de este dispositivo no se deja cambiar a modo monitor, por lo tanto fue necesario implementarle la siguiente antena USB:

|

Ilustración 8 Instalación de kismet, antena

Configuración de la antena: Para cambiar la tarjeta a modo monitor se escriben los siguientes comandos en la consola:

- **sudo ifconfig wlan0 down**
- **sudo iwconfig wlan0 mode monitor**
- **sudo ifconfig wlan0 up**

***La interface es a la que está conectada la antena, para mirar escriba el comando: ifconfig

Configuración de kismet:

Vamos al archivo de configuración de kismet kismet.conf que generalmente se encuentra en la ruta /etc/kismet/kimet.conf y la abrimos con el siguiente comando:

- **nano kismet.conf**

Añadimos al final del archivo “ncsource = wlan0” sin las “”, para que kismet reconozca por de interface se va hacer el escaneo.

Corriendo Kismet:

Para correr kismet ese escribe el comando:

- **kismet**

Se abrirá una ventana de kismet:

Ilustración 9 Instalación de kismet,

Si queremos ver la información por la consola de kismet le damos click en **[]Show Console**

Le damos click en **Start.**

Y kismet empieza a monitorear las redes WIFI.

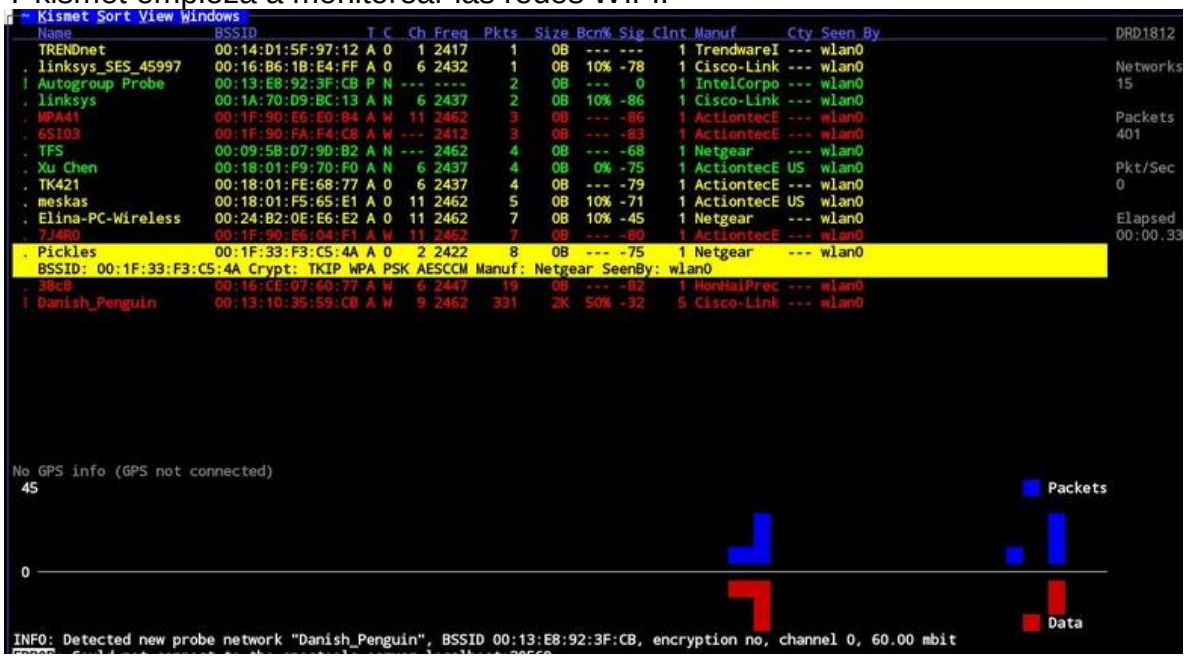


Ilustración 10 Instalación de kismet, ventana de monitoreo

CONFIGURACIÓN KISMET EN OSSIM

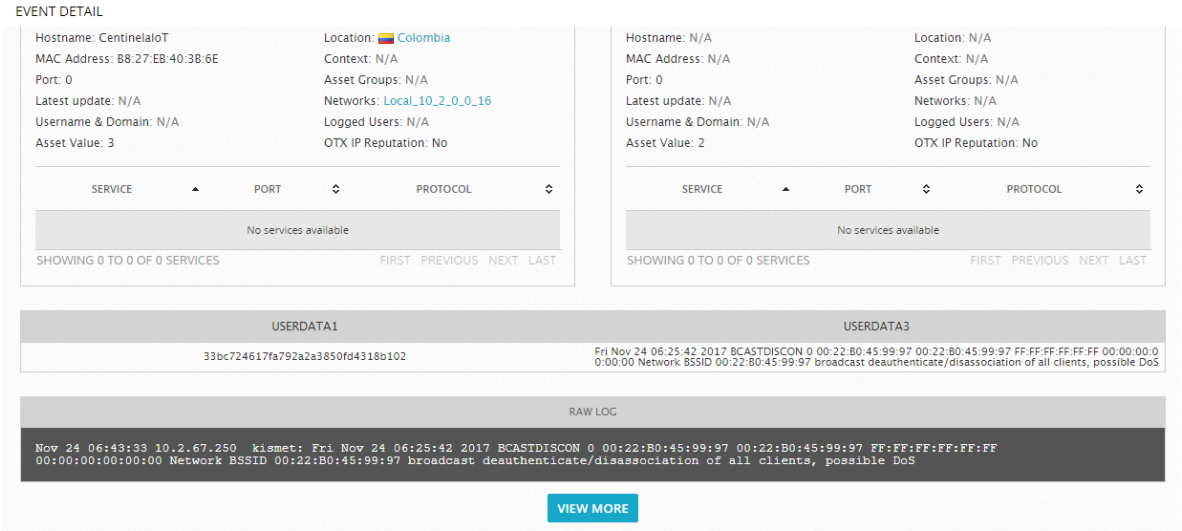
Para cuando Kismet genere la alerta generalmente la guarda en la siguiente ruta **etc/kismet** y la envia al OSSIM y para esto se debe tener corriendo el programa que está en el siguiente Github https://github.com/andresvega82/SIEM-IoT/tree/master/Software/Kismet
en el archivo Syslogkismet.zip, el cual envía el syslog al OSSIM para que este pueda recibir el evento.
En la siguiente imagen podemos ver en color amarillo el código donde se envía el syslog:

```
        // Initialise sender
UdpSyslogMessageSender messageSender = new UdpSyslogMessageSender();
        messageSender.setDefaultMessageHostname(""); // some syslog cloud services may use this field to t
        messageSender.setDefaultAppName("kismet");
        messageSender.setDefaultFacility(Facility.USER);
        System.out.println("lo que tiene USER"+Facility.USER.name());
        messageSender.setDefaultSeverity(Severity.ALERT);
        //messageSender.setSyslogServerHostname("10.2.78.8");
        messageSender.setSyslogServerHostname("192.168.0.5");
        messageSender.setSyslogServerPort(514);
        messageSender.setMessageFormat(MessageFormat.RFC_3164);

         try {
      FileReader fr = new FileReader(fichero);
      BufferedReader br = new BufferedReader(fr);

      String linea;
      while((linea = br.readLine()) != null){
          lineas.add(linea);
          //System.out.println(linea);
      }

      for(int i= 0; i<lineas.size(); i++){
           System.out.println(lineas.get(i));
          // send a Syslog message

          try{
              messageSender.sendMessage(lineas.get(i));
                  System.out.println("el mensaje enviado fue:"+ lineas.get(i));}catch (IOException ex) {
        Logger.getLogger(ex.getMessage());
        }
```

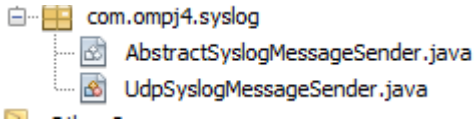
Ilustración 11 configuración de kismet en Ossim, envio del syslog

Este programa lee los nuevos archivos que se van creando en la carpeta ect/kismet y los envía al ossim.

La forma en que llega el evento al OSSIM es la siguiente:



**Es necesario incluir las clases:



**Para correr el programa se genera un .jar
** y se ejecuta con el conmando **java -jar <el nombre del archivo >.Jar.**

Ya con lo anterior se envía el syslog al OSSIM , pero para indicarles que campos del syslog queremos que se vea en el evento, es necesario agregar en la plataforma de OSSIM le plugin kismetIoTPlugin.cfg, el cual se encuentra en el Github mencionado anteriormente, el cual contiene la expresión regular:

```
[syslog - KimsteSyslog]
event_type=event
regexp="([\s\S]+) (?P<src_ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})  (kismet: )([\s\S]+)(?P<date>\w+ \d\d \d\d:\d\d:\d\d)([\s\S]+)(BCASTDISCON
plugin_sid=1
src_ip={resolv($src_ip)}
username={$date}
userdata2={$channel}
userdata3={$src_mac}
userdata4={$dest_mac}
userdata5={$chan}
```

Ilustración 12 configuración de kismet en Ossim, gluginkismet

Continuación de la expresión regular

```
(?P<channel>\d+) (?P<src_mac>\w\w(:)\w\w(:)\w\w(:)\w\w(:)\w\w(:)\w\w) (?P<dest_mac>\w\w(:)\w\w(:)\w\w(:)\w\w(:)\w\w(:)\w\w)(?P<chan>[\s\S]+)"
```

Ilustración 13 configuración de kismet en Ossim

La cual hará que el OSSIM reciba y muestre el evento de la siguiente forma:

```
Nov 25 11:45:57 10.2.67.240  -: Nov 25 16:10:53 10.2.67.240  kismet: Sat Nov 25 16:10:53 2017 BCASTDISCON 0 90:F6:52:89:7C:B2 90:F6:52:89:7C:B2
FF:FF:FF:FF:FF:FF 00:00:00:00:00:00 Network BSSID 90:F6:52:89:7C:B2 broadcast deauthenticate/disassociation of all clients, possible DoS
```

Ilustración 14 configuración de kismet en Ossim, evento de kismet

PRUBAR KISMET

Generar una alerta en Kismet:
En este escenario vamos a generar la alerte **BCASTDISCON** la cual se dispara cuando detecta que hay un ataque de desasociación de la red de un cliente o de varios, causando una posible denegación de servicio.
Debemos tener el Kismet corriendo:
Comando : **Kismet**
Para generar el ataque vamos a utilizar la herramienta **aireplay-ng,** en nuestro caso vamos a desconectar a todos los clientes conectados a la red:
Escribimos el siguiente comando:

- **aireplay-ng –deauth 0 –a < BSSID> wlan1**

  *el 0(cero) indica que va a realizar el ataque indefinidamente, si queremos que solo se realice 1 vez, cambiamos el 0 por 1.
  * wlan1 es la interface de red.
  * El BSSID es el punto de acceso por el cual se va a realizar el ataque.

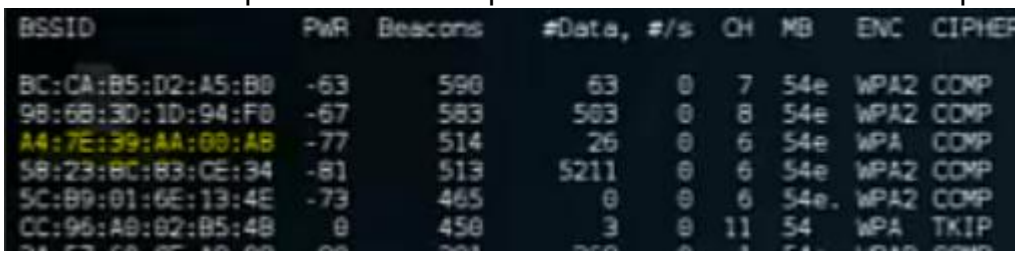


Ilustración 15 prueba kismet, detectar las BssiD que están en la red.

Cualquiera que se encuentre en la red para visualizar mejor estas BSSID podemos utilizar el comando: **airodump-ng wlan1**, utilizando la interface de red que tengamos en el momento.

Luego le damos Enter y se empieza el ataque, se demora un poco ya que tiene que coincidir el canal(CH) del BSSID con el de la interface de red(wlan1).

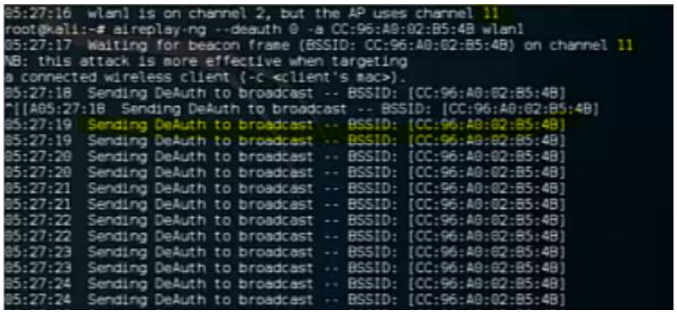


Ilustración 16 prueba kismet, paquetes que envía el ataque

En la imagen anterior ponemos ver que empieza a enviar paquetes de desasociación por el punto que acceso que definimos y de este modo generar una denegación de servicio.
Kismet detecta el ataque:

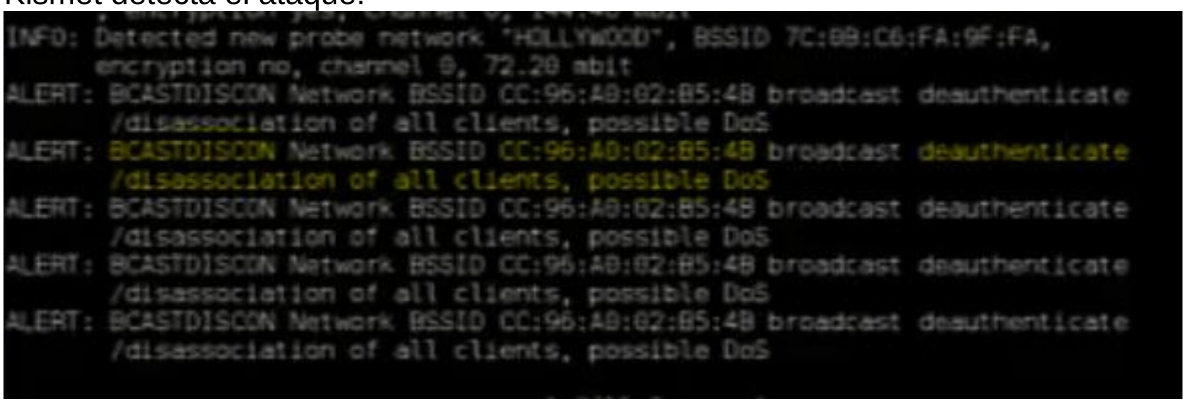


Ilustración 17 prueba kismet, alertas detectadas por kismet

Informado que tipo de ataque es, por donde se está generando el ataque y cual la consecuencia de este.
Cuando kismet recibe la alerta esta es reportada al OSSIM y el OSSIM por medio del siguiente script, realiza una acción:

```
import paramiko
import sys
def sshCommand(hostname,port,username,password,command):
    sshClient = paramiko.SSHClient()

    sshClient.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    sshClient.load_system_host_keys()
    sshClient.connect(hostname,port,username,password)
    stdin, stdout, stderr = sshClient.exec_command(command)
    print(stdout.read())

if __name__=='__main__':
    sshCommand(sys.argv[0],22,'root','toor','reboot')
```

Ilustración 18 prueba kismet scripts: respuesta al ataque

En nuestro caso lo que hace el reiniciar el equipo cerrando el punto de acceso del atacante (se debe tener en cuenta que para este paso se deben tener permisos de administrador). En el OSSIM se programa de la siguiente manera:

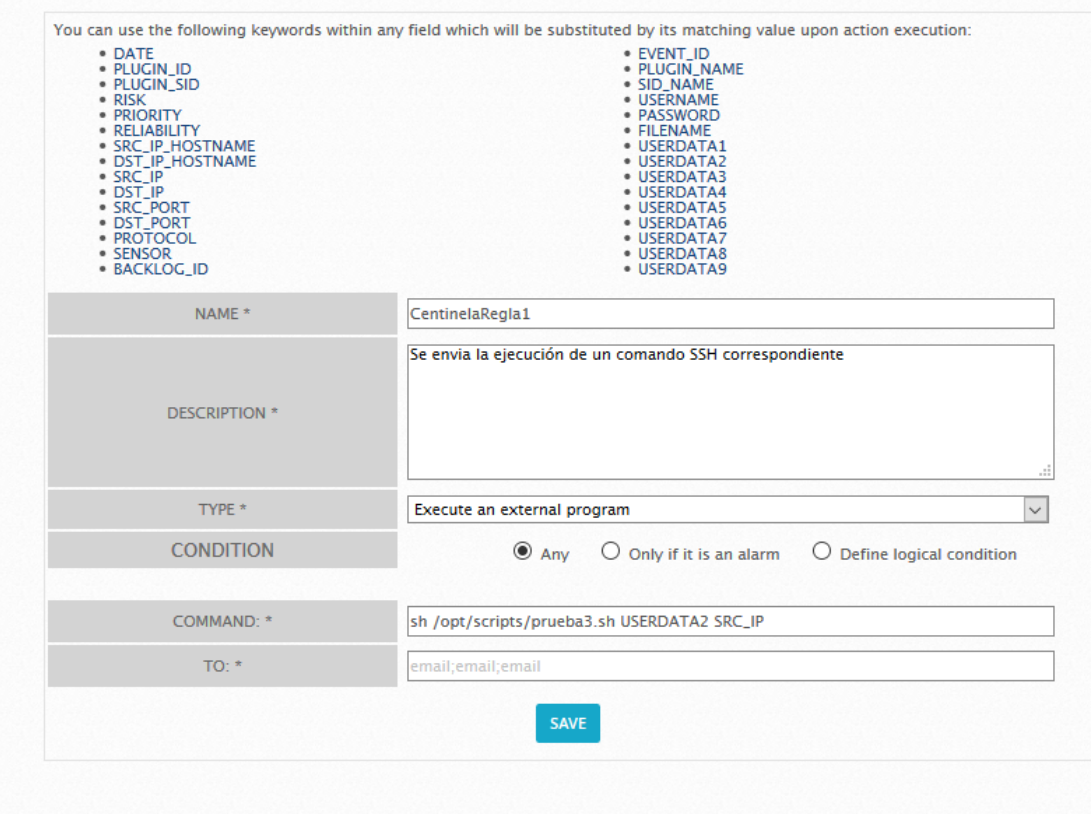


Ilustración 19 prueba kismet, como se configura el scripts en Ossim

|

COMMAND:* se indica el scripts que se va a utilizar.
En TO* se puede poner un correo para informar al administrador de que ocurrió el evento.

6.2 OPENVAS

OpenVas es una agrupación de herramientas para realizar la gestión, escaneo y análisis de vulnerabilidades a través de la red sobre diferentes dispositivos., esto es basado en NVT's (Network Vulnerabilty Tests) lanzados sobre targets definidos en la interfaz web de la herramienta construida y mantenida por Greenbone. También tiene servicios para detección de sistemas operativos y detección de dispositivos conectados en la red que se encuentre conectado el OpenVas.

Instalación OpenVas:

1. apt-get update
2. apt-get dist-upgrade
3. apt-get install openvas = Este paso podría tomar bastante tiempo.
4. openvas-setup = Retornada la creación del usuario admin con su correspondiente clave.
5. netstat -antp = verificar el servicio
6. openvas-start
7. openvas-check-setup = Retornara un mensaje "OpenVas Installation OK".
8. Abrir esta dirección url para comprobar que el servicio este disponible https://127.0.0.1:9392 = Ingresar con el usuario admin anteriormente mencionado y cambiar la clave.

Archivo Plugin para OSSIM

El archivo de plugin para OSSIM es el archivo llamado openVasPlugin.cfg, este archivo contiene las especificaciones de la expresión regular que permite al OSSIM entender los eventos generados por esta herramienta enviados por el código OMP4-OpenVas.

La expresión regular para OSSIM es:

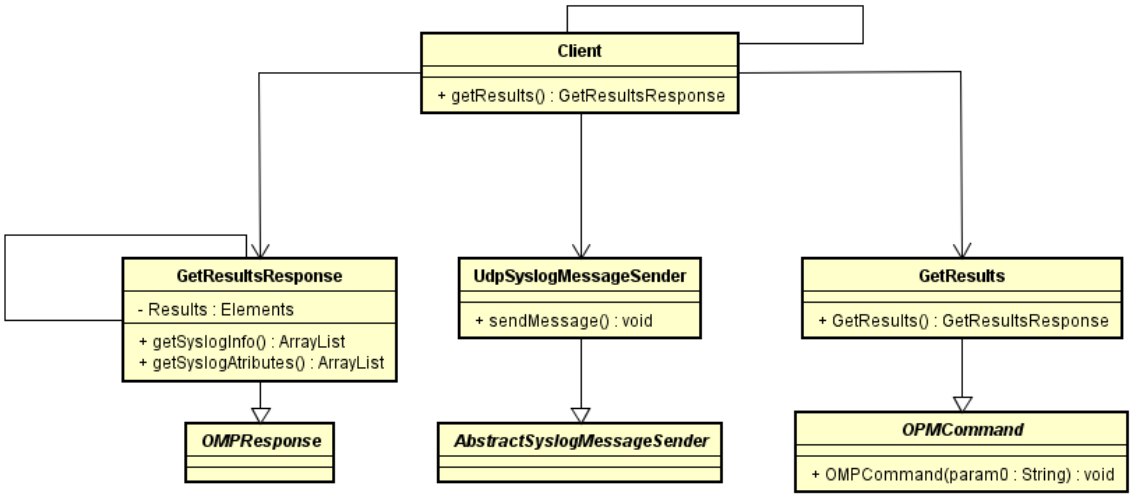
"([\s\S]+)(OpenVas:
)(|)(?P<vul_id>[\s\S]+)(|)(?P<ip_address>[\s\S]+)(|)(?P[\s\S]+)(|)(?P[\s\S]+)"

La cual inicia con un indicador "OpenVas: " que nos indica que es un mensaje de la herramienta, luego de esto vienen 4 campos distintos separados por "|" el primer campo es el id de la vulnerabilidad, segundo campo el ip de la fuente, tercer campo severidad de la vulnerabilidad y en el último campo el cve asociado con la vulnerabilidad.

Código OpenVasOMP:

El código del OpenVasOMP cuenta con diversas clases java para realizar peticiones que el protocolo OMP nos permite para interactuar con OpenVas en este caso estamos utilizando consultas xml desde el código para extraer los resultados y ser enviados al OSSIM.

Diagrama de clases OpenVasOMP



La clase principal es Client la encargada de realizar todo el proceso, lo primero que hace es utilizar un método de si misma “getResults” el cual crea una instancia de “GetResults” en donde se identifica el comando necesario para obtener los resultados directamente del OpenVas, cuando genera el comando este retorna un código xml el cual es usado para crear una nueva instancia de la clase “GetResultsResponse”.

Una vez que tenemos esta instancia creada es retornada a Client y se procede a utilizar el método “getSysloginfo” quien se encarga de separar cada resultado identificado en el xml y a su vez utiliza el método “getSyslogAtributes” quien se encarga de obtener los atributos deseados de cada resultado respectivamente y se organiza en ArrayList para ser enviado al OSSIM.

Cuando Client ya obtiene el ArrayList de cada resultado con sus respectivos atributos, procede a crear un mensaje syslog por resultado del ArrayList obtenido

con la clase “UdpSyslogMessageSender” y a través del método “sendMessage” el cual enviara el mensaje syslog a la dirección establecida desde la clase Client.

Prueba OpenVas:

Una vez que hemos iniciado sesión como administradores (como se dice en el manual de instalación) nos dirigiremos a targets y alli creamos uno nuevo con la ip que deseemos escanear mientras se encuentre en la misma red que el OpenVas, seguido de esto en task, crearemos un nuevo task apuntando al target anteriormente creado y le damos start, si funciona correctamente el deberá iniciar una carga porcentual que tardara un tiempo dependiendo que tan invasivo será el escaneo, al final de esta carga podremos ver todas las vulnerabilidades encontradas en el target, clasificadas en 4 severidades diferentes (Informativas, bajas, media , alta)

Para correr Codigo OMP4-OpenVas

Este código fue creado con el fin de enviar todos los resultados hacia OSSIM de cada análisis realizado por OpenVAS utilizando el API de la herramienta llamada OpenVAS Management Protocol (**OMP**).

1. Utilizando contrab creamos un task en Linux para ejecutar el código de la siguiente forma "contrab -e"
2. Luego escribimos el comando estableciendo cuando y a que hora deberá ejecutarlo en este será todos los días a las 3 de la mañana con el siguiente código "0 3 * * * "path del código" java -jar OPM4-OpenVas.jar"

6.3 SURICATA IOT

Suricata IoT es un sistema para la detección de intrusiones (IDS) con una modificación particular de un proyecto creado por Tom DeCanio que permite realizar detección de tráfico malicioso de algunos protocolos presentes en arquitecturas de dispositivos IoT [22].

Suricata IoT al ser una modificación del sistema original de Suricata, está también obtiene las diferentes características que tiene el sistema de Suricata, como lo es la capacidad de ser un sistema de detección de intrusiones (IDS), ser un sistema de prevención de intrusiones (IPS) o monitoreo de seguridad de red (NSM).

Suricata IoT utiliza el monitoreo de trafico de red basado en reglas robustas para identificar ataques, y estas reglas son administradas mediante un proveedor de

reglas, en este caso se trata de OinkMaster, sin embargo, no es el único, existen otras fuentes o se puede construir reglas propias que permitan identificar ataques sobre protocolos de la capa de red, aplicación y demás.

Cabe resaltar que Suricata es un sistema de código abierto, por lo tanto permite realizar modificaciones como la que ha realizado Tom DeCanio para la extensibilidad de esta herramienta y aumentar su potencial para nuevas arquitecturas.

Instalación Suricata:

1. Lista que librerías para instalar sobre SO:

```
sudo apt-get -y install libpcre3 libpcre3-dbg libpcre3-dev \
build-essential autoconf automake libtool libpcap-dev libnet1-dev \
libyaml-0-2 libyaml-dev pkg-config zlib1g zlib1g-dev libcap-ng-dev libcap-ng0 \
make libmagic-dev libjansson-dev

wget https://github.com/OISF/libhtp/archive/0.5.21.tar.gz
tar -xzvf 0.5.21.tar.gz
cd libhtp
./autogen.sh
./configure
make
make install
ldconfig
```

2. Descargar comprimido de Suricata:

```
git clone https://github.com/decanio/suricata-IoT.git
```

3. Entrar a la carpeta de suricata:

```
cd suricata-3.1
```

4. Instalar:

```
./autogen.sh
./configure && make && make install-full
ldconfig
```

5. Copiar archivos de configuración:

```
mkdir /var/log/suricata
mkdir /etc/suricata
mkdir /etc/suricata/rules
cp classification.config /etc/suricata
cp reference.config /etc/suricata
cp suricata.yaml /etc/suricata
```

6. Habilitar el envió de alertas al servicio syslog para enviar alertas a OSSIM, editar archivo “suricata.yaml”:

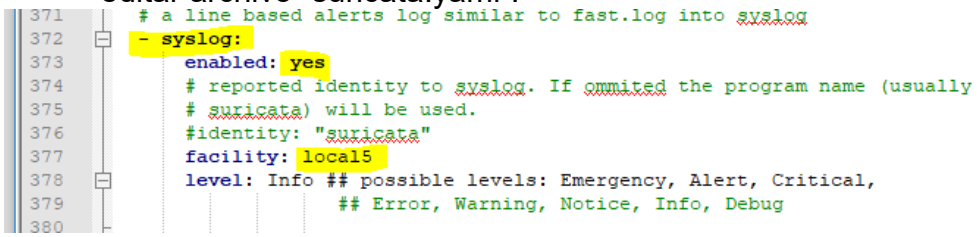


Ilustración 20 Archivo de configuración suricata

7. Incluir el envió de alertas de Suricata que tiene como prefijo “local5” por el servicio Syslog, incluir la siguiente línea en el archivo /etc/rsyslog.conf:

*.local5@ip_servidor_OSSIM:puerto

8. Reiniciar servicio rsyslog:

sudo service rsyslog restart

9. Bajar reglas (por defecto se bajan por Oinkmaster):

apt-get install oinkmaster
editar el archivo oinkmaster.conf: /etc/oinkmaster.conf
adicionar línea:
url = http://rules.emergingthreats.net/open/suricata/emerging.rules.tar.gz
comando: oinkmaster -C /etc/oinkmaster.conf -o /etc/suricata/rules

10. Correr Suricata:

suricata -c /etc/suricata/suricata.yaml -i eth0 --init-errors-fatal

Archivo configuración suricata.yaml

El archivo de configuración de Suricata IoT es el archivo llamado suricata.yaml, este contiene los parámetros para correr el suricata, las partes principales de este archivo están en la configuración de la red, en donde se identifica la red local y la red externa, y las reglas que se quieren aplicar.

```
## Step 1: inform Suricata about your network
##

vars:
  # more specifc is better for alert accuracy and performance
  address-groups:
    #HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]"
    HOME_NET: "[192.168.0.0/24]"
    #HOME_NET: "[10.2.67.0/16]"
    #HOME_NET: "[172.16.0.0/12]"
    #HOME_NET: "any"

    EXTERNAL_NET: "!$HOME_NET"
    #EXTERNAL_NET: "any"

    HTTP_SERVERS: "$HOME_NET"
    SMTP_SERVERS: "$HOME_NET"
    SQL_SERVERS: "$HOME_NET"
    DNS_SERVERS: "$HOME_NET"
    TELNET_SERVERS: "$HOME_NET"
    AIM_SERVERS: "$EXTERNAL_NET"
    DNP3_SERVER: "$HOME_NET"
    DNP3_CLIENT: "$HOME_NET"
    MODBUS_CLIENT: "$HOME_NET"
    MODBUS_SERVER: "$HOME_NET"
    ENIP_CLIENT: "$HOME_NET"
    ENIP_SERVER: "$HOME_NET"
```

Ilustración 21 Archivo de configuración suricata 1

En esta parte del archivo de configuración suricata.yaml sirve para configurar las propiedades de la red en donde está el dispositivo centinela, en tal caso, se define la variable de “HOME_NET”, en donde se coloca el identificador de red y la máscara, como lo muestra la imagen, de igual modo, se puede configurar otro tipo de variable en donde se encuentran algunos servicios de red como el servidor HTTP o el DNS.

```
## Step 2: select the rules to enable or disable
##

default-rule-path: /etc/suricata/rules
rule-files:
# - app-layer-events.rules
# - decoder-events.rules
# - dns-events.rules
# - files.rules
# - http-events.rules
# - modbus-events.rules
# - smtp-events.rules
# - stream-events.rules
# - tls-events.rules
# - mqtt-events.rules
 - emerging-dos.rules
# - emerging-exploit.rules
# - emerging-ftp.rules
# - emerging-games.rules
# - emerging-icmp_info.rules
# - emerging-icmp.rules
# - emerging-imap.rules
# - emerging-inappropriate.rules
# - emerging-malware.rules
# - emerging-misc.rules
# - emerging-mobile_malware.rules
# - emerging-netbios.rules
```

Ilustración 22 Archivo de configuración suricata 2

En este punto del archivo de configuración suricata.yaml permite escoger las reglas que van a ser revisadas por el software Suricata, en este punto se debe ver que cada archivo de reglas tiene un esquema de nombre_del_archivo.rules, en donde el nombre describe el paquete de reglas de se evalúan. Por otro lado, también se configura la ubicación en donde se encuentran las reglas, esta configuración se ve en la asignación de la variable “default-rule-path”.

```
## Step 3: select outputs to enable
##

# The default logging directory.  Any log or output file will be
# placed here if its not specified with a full path name. This
can be
# overridden with the -l command line parameter.
default-log-dir: /var/log/suricata/

# global stats configuration
stats:
  enabled: yes
  # The interval field (in seconds) controls at what interval
  # the loggers are invoked.
  interval: 86400

# Configure the type of alert (and other) logging you would
like.
outputs:
  # a line based alerts log similar to Snort's fast.log
  - fast:
      enabled: yes
      filename: fast.log
      append: yes
      #filetype: regular # 'regular', 'unix_stream' or
'unix_dgram'
```

Ilustración 23 Archivo de configuracón suricata 3

Por último, en esta parte del archivo de configuración suricata.yaml permite configurar las diferentes salidas de reportes de alertas o avisos que emite el sistema, como muestra la imagen permite configurar un archivo de estadísticas, al colocar el campo de “enables” en “yes” se toma que se quiere un archivo de reporte de estadísticas cada cierto tiempo definido en la variable “Interval”. Todos estos archivos de reportes se pueden consultar en la ruta colocada en la variable de “default-log-dir”.

Archivo Plugin para OSSIM

El archivo de plugin para OSSIM es el archivo llamado SuricataIoT.cfg, este archivo contiene las especificaciones de la expresión regular que permite al OSSIM entender los eventos generados por esta herramienta.

```
[syslog - IDSSuricata]
event_type=event
regexp="(?P<date>\w+ \d\d \d\d:\d\d:\d\d)([\s\S]+)(\[)(?P<ext>\d+)(\])(:) (\[)(?P<pid>\d+)(:)(?P<sid>\d+)(:)(\d+)(\]) (?P<msg>
plugin_sid=1
src_ip={$src_ip}
src_port={$src_port}
dst_ip={$dst_ip}
userdata1={$dst_port}
userdata2={$sid}
protocol={$proto}
priority={$prio}
date={$date}
```

Ilustración 24 Archivo plugin suricata

Con este archivo de plugin el sistema OSSIM reconoce los mensajes del protocolo syslog que tienen específicamente el formato determinado por la expresión regular ya mencionada, de tal forma que nos permite obtener información importante para determinar el tipo de tráfico malicioso, para ello OSSIM permite declarar variables tomadas de la expresión regular las cuales son las siguientes:

- src_ip, esta variable contiene la ip fuente del tráfico identificado como malicioso.
- src_port, esta variable contiene el puerto fuente del tráfico malicioso.
- dst_ip, esta variable contiene la ip de destino del tráfico malicioso.
- userdata_1, esta variable contiene el puerto destino del tráfico malicioso.
- userdata_2, esta variable contiene el número de identificación de la regla Suricata que se envía.
- priority, esta variable contiene el número de prioridad definida en la regla Suricata del tráfico malicioso.

Prueba Suricata:

1. Para probar Suricata corremos el comando en la terminal:
   suricata -c /etc/suricata/suricata.yaml -i eth0 --init-errors-fatal

2. Una vez comience a correr el mismo sistema empieza a examinar los paquetes de la red en busca que coincida con alguna regla.

3. De haber un paquete que coincida con alguna regla de Suricata y se genere una alerta Suricata, el sistema automáticamente lo enviará al servidor OSSIM y lo entenderá como un mensaje de alerta Suricata. Lo anterior se logra gracias a que se configuro que toda alerta sea dirigida al servicio rsyslog y que se envíe al sistema OSSIM como un mensaje Syslog (Ver proceso de instalación de suricata en los pasos 6,7 y 8).

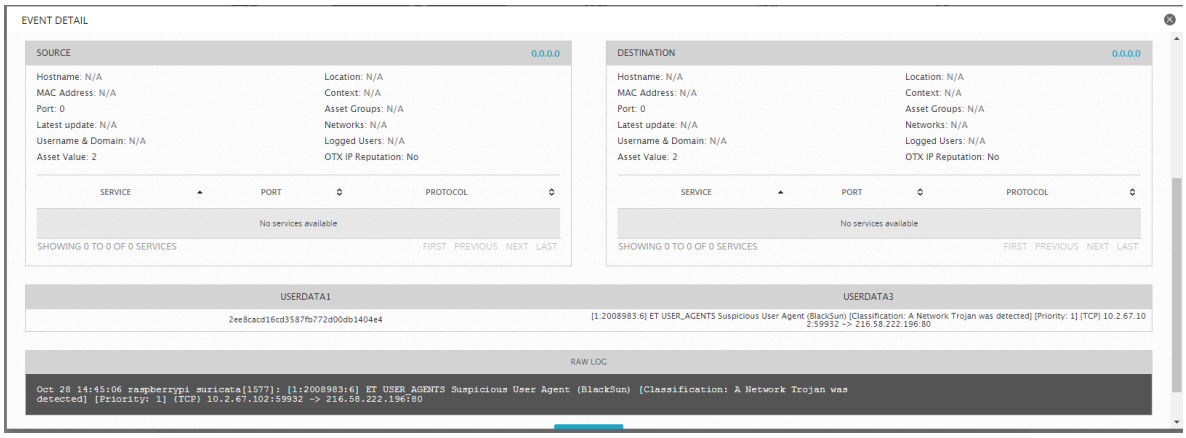


Ilustración 25 Evento suricata en OSSIM

7. ARQUITECTURA CENTINELA

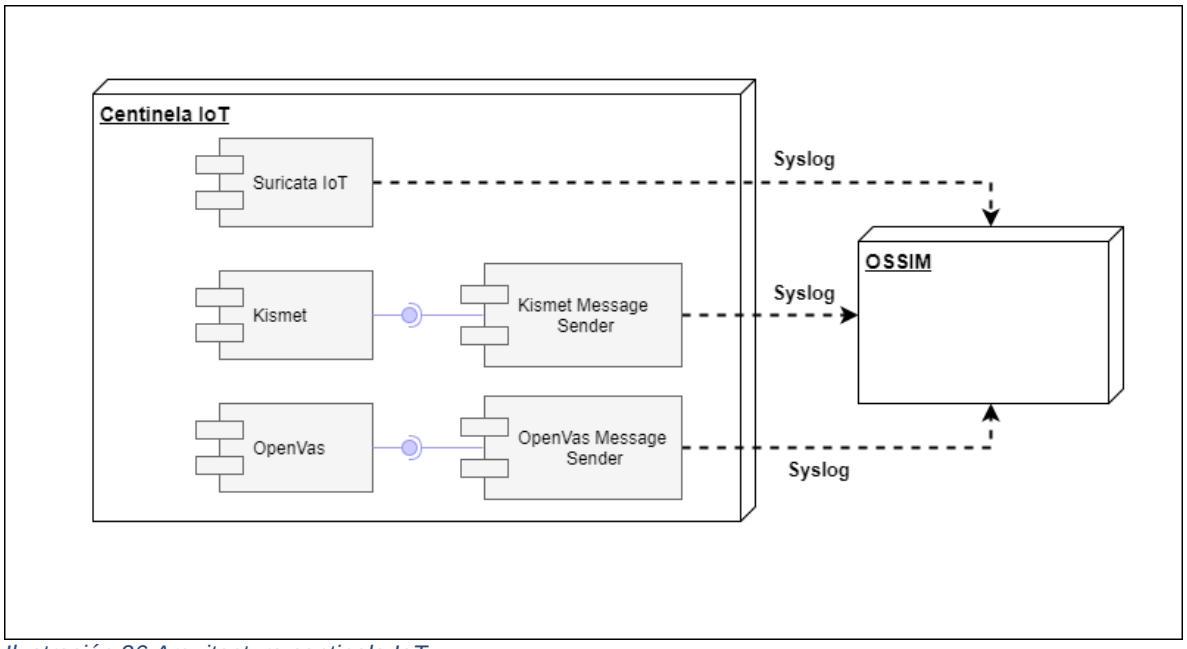


Ilustración 26 Arquitectura centinela IoT

En la arquitectura de la solución propuesta del dispositivo centinela se cuenta con 3 módulos: Suricata IoT, Kismet y OpenVas; estos módulos representan cada herramienta que se usa para brindar protección a los dispositivos al alcance del centinela.

En primer lugar, el módulo de Suricata IoT, es una herramienta que permite utilizar reglas propias del software Suricata IDS (Sistema de Detección de Intruciones) con una modificación para la detección de tráfico de red propios de dispositivos IoT, cabe resaltar que este software solo sirve para monitorear el tráfico de red ethernet.

El segundo módulo es el de Kismet, que tiene cierta similitud con Suricata, es un IDS (Sistema de Detección de Intrusiones) especializado para detectar el tráfico de paquetes en redes inalámbricas, por ello, este software tiene un conjunto de reglas de detección de ataques especializado para redes inalámbricas.

Por último, el tercer módulo llamado OpenVas, es un software que permite analizar vulnerabilidades de los diferentes dispositivos basados en una base de datos actualizada de vulnerabilidades de diferentes dispositivos, entre los cuales, aplican los dispositivos IoT.

Los otros dos módulos restantes son para enviar eventos de seguridad a la plataforma de OSSIM para que puedan procesarse y generar las reglas de correlación que en el siguiente capítulo se explicaran.

8. DIRECTIVAS DE CORRELACIÓN Y RESPUESTAS

La **primera directiva** trata de tener dos eventos, unos de openvas y otro de suricata, el primer evento es la vulnerabilidad(CVE-2012-5964,ST URN ServiceType Buffer Overflow) de la librería libupnp que es vulnerable a un ataque de denegación de servicio por medio de un mensaje del protocolo ssdp en donde el campo de service type de ese mensaje tiene un valor muy grande, y el segundo evento trata de un evento de suricata en donde identificar tráfico malicioso de un mensaje ssdp hacia el dispositivo upnp en donde se evidencia ciertas palabras claves que dan como positivo el ataque de denegación de servicio del dispositivo. El SIEM como respuesta a estos eventos realizará una actualización de la librería libupnp.

Esta directiva se puede probar en un dispositivo IoT uPnP que tenga la librería libupnp en su versión 1.3.1, en donde el dispositivo centinela con ayuda de Openvas detecta el uso de esta librería vulnerable, luego con el monitoreo constante de la red de la herramienta de Suricata IoT, se debe detectar tráfico malicioso, en donde un mensaje del protocolo SSPD (En un paquete UDP) con el campo de "ServiceType" presenta un tamaño muy grande.

Una vez la herramienta OSSIM recibe los eventos generados por las herramientas Openasvas y Suricata IoT, este debido a su configuración de la directiva de correlación genera una respuesta ejecutado un script en el dispositivo centinela, que a su vez ejecuta un script que actualiza la librería vulnerable del dispositivo que está siendo atacado.

El orden de la ejecución de esta directiva queda de esta manera:

1. El dispositivo centinela detecta la vulnerabilidad asociada con el código de CVE-2012-5964.

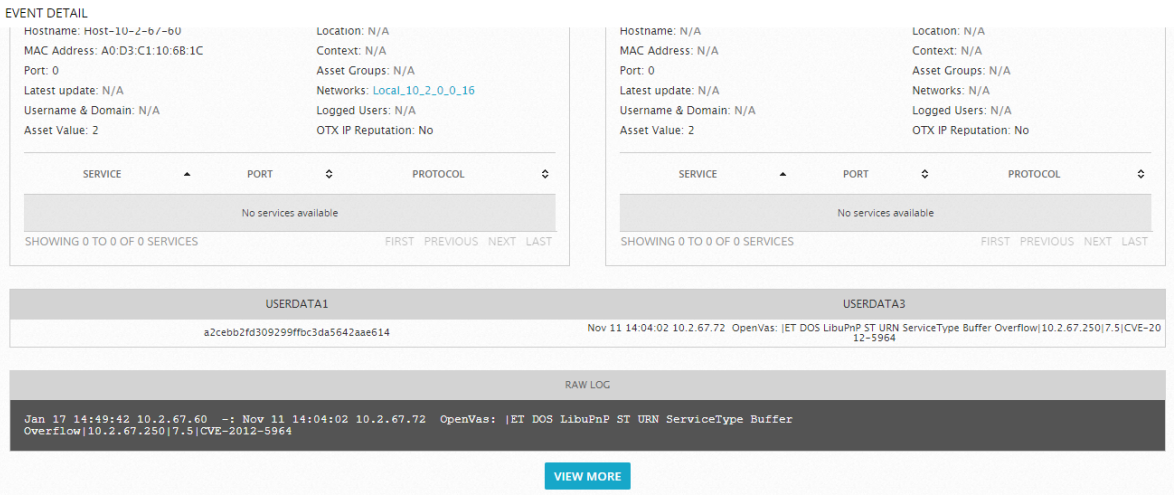


Ilustración 27 Evento Openvas directiva 1

2. El dispositivo centinela detecta tráfico malicioso y lo envía a la plataforma de OSSIM.

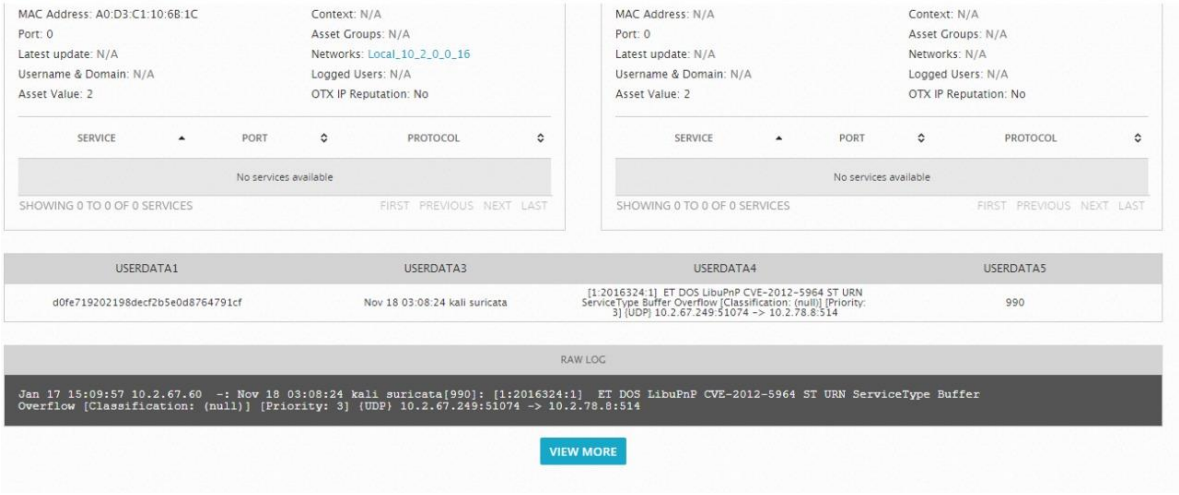


Ilustración 28, Evento Suricata directiva 1

3. Gracias a la configuración de OSSIM se genera la correlación cruzada de eventos, esta consiste en tener diferentes fuentes de eventos de seguridad reportando que permiten inferir ataques de seguridad, en el caso de esta directiva de genera un evento de Openvas y otro evento de Suricata, lo cual al tener eventos de estas dos fuentes se activa la directiva de correlación cruzada. Cabe resaltar que para que se active esta directiva de correlación los eventos que llegan deben ser del mismo tipo, tanto la vulnerabilidad específica y el ataque.

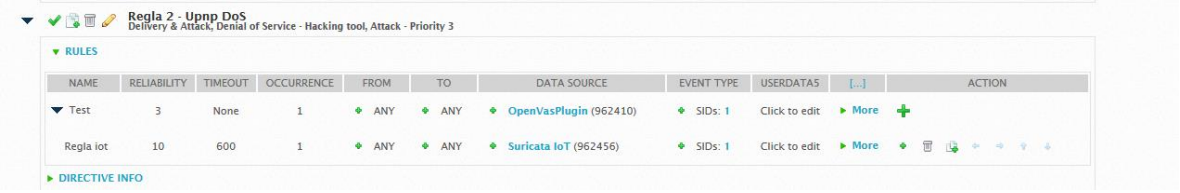


Ilustración 29 Regla de correlación 1

4. La plataforma OSSIM detecta el ataque y genera una respuesta de contingencia al ataque, para este caso se actualiza la librería libupnp del dispositivo atacado.

```
import paramiko
import sys
def sshCommand(hostname,port,username,password,command):
    sshClient = paramiko.SSHClient()

    sshClient.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    sshClient.load_system_host_keys()
    sshClient.connect(hostname,port,username,password)
    stdin, stdout, stderr = sshClient.exec_command(command)
    print(stdout.read())

if __name__=='__main__':
    sshCommand(sys.argv[0],22,'root','toor','apt-get install libupnp-dev')
```

Ilustración 30, script de actualización libupnp

Para el caso de esta respuesta activa, cabe resaltar que se debe tener permisos de administrador del dispositivo que se va a realizar la operación de actualización en el dispositivo centinela para que esta respuesta funcione.

La **segunda directiva** trata de la denegación de servicio de un dispositivo que tenga un servicio web disponible, basado en el uso de un servicio de Nginx. El modo de operar es el mismo que el anterior, primero se tiene la vulnerabilidad (CVE-2013-2028, Exploit Specific) del dispositivo que dice que la versión del servicio Nginx es vulnerable a ataques de denegación de servicio, el segundo evento es la evidencia de tráfico malicioso que da a entender que se está explotando la vulnerabilidad ya mencionada mediante una petición al dispositivo con unos campos específicos. El SIEM como respuestas a estos eventos se genera la instalación de nginx.

La segunda directiva se prueba de tal forma que un dispositivo IoT use la librería Ngix en su versión 1.3.9 hasta la versión 1.4.0, lo cual la herramienta de Openvas detecta el uso de esta librería vulnerable, luego con el monitoreo constante de la red con la herramienta de Suricata IoT, se detecta una petición HTTP en donde el paquete tiene como encabezado “Transfer-Encoding: chunked”.

Una vez la herramienta OSSIM recibe los eventos generados por las herramientas Openasvas y Suricata IoT, este debido a su configuración de la directiva de correlación genera una respuesta ejecutado un script en el dispositivo centinela, que a su vez ejecuta un script que actualiza la librería vulnerable del dispositivo que está siendo atacado.

El orden de la ejecución de esta directiva queda de esta manera:

1. El dispositivo centinela detecta la vulnerabilidad asociada con el código de CVE-2012-5964.

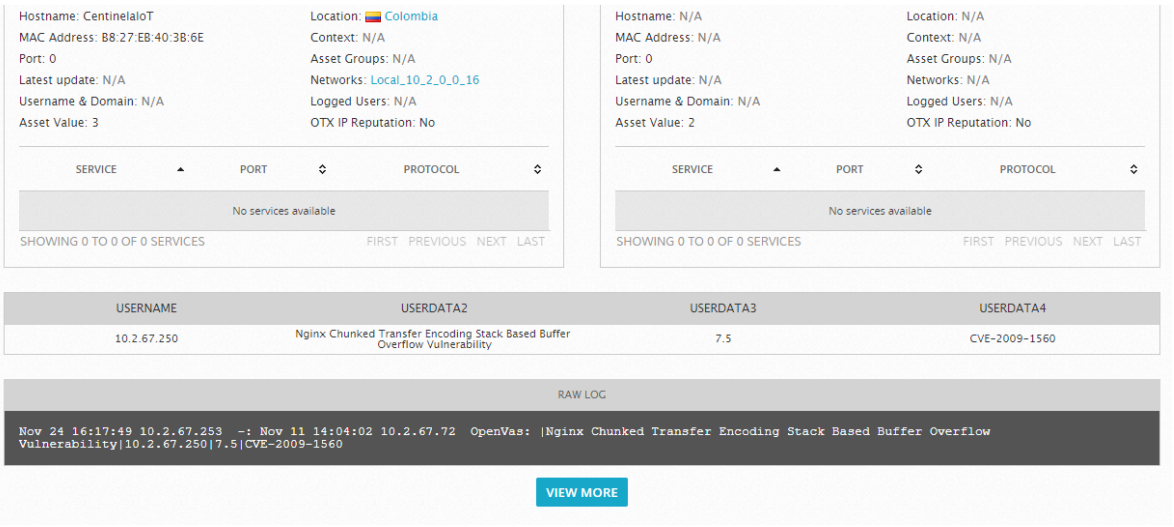


Ilustración 31 Evento Openvas directiva 2

2. El dispositivo centinela detecta tráfico malicioso y lo envía a la plataforma de OSSIM.

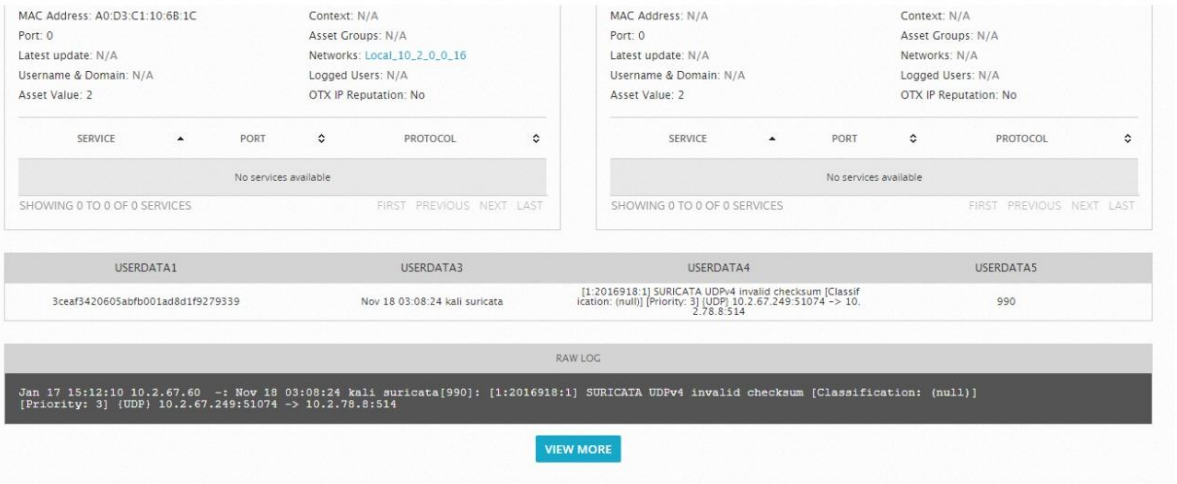


Ilustración 32 Evento Suricata directiva 2

3. Gracias a la configuración de OSSIM se genera la correlación cruzada de eventos, esta consiste en tener diferentes fuentes de eventos de seguridad reportando que permiten inferir ataques de seguridad, en el caso de esta directiva de genera un evento de Openvas y otro evento de Suricata, lo cual al tener eventos de estas dos fuentes se activa la directiva de correlación cruzada. Cabe resaltar que para que se active esta directiva de correlación los eventos que llegan deben ser del mismo tipo, tanto la vulnerabilidad específica y el ataque.

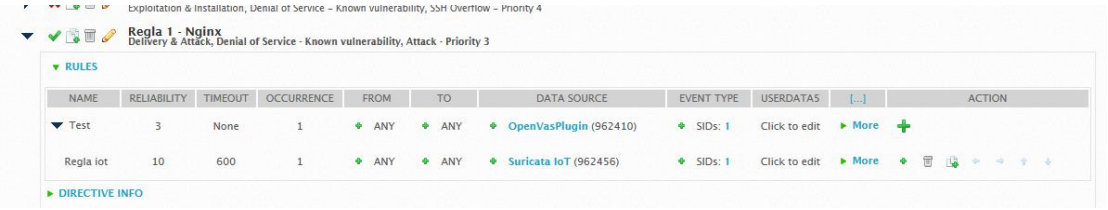


Ilustración 34 Directiva de correlación cruzada 2

4. La plataforma OSSIM detecta el ataque y genera una respuesta de contingencia al ataque, para este caso se actualiza la librería Nginx del dispositivo atacado.

```
import paramiko
import sys

def sshCommand(hostname,port,username,password,command):
    sshClient = paramiko.SSHClient()

    sshClient.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    sshClient.load_system_host_keys()
    sshClient.connect(hostname,port,username,password)
    stdin, stdout, stderr = sshClient.exec_command(command)
    print(stdout.read())

if __name__=='__main__':
    sshCommand(sys.argv[0],22,'root','toor','apt-get install nginx')
```

Ilustración 35, script de actualización Nginx

Para el caso de esta respuesta activa, cabe resaltar que se debe tener permisos de administrador del dispositivo que se va a realizar la operación de actualización en el dispositivo centinela para que esta respuesta funcione.

La **tercera directiva** de correlación se basa en una vulnerabilidad(CVE: CVE-2017-13077) sobre el protocolo WPA(Acceso protegido Wi-Fi) analizada por OpenVas (la cual se encuentra en el sistema operativo DEBIAN con versión 2.3-1) y la alerta de kismet llamada BCASTDISCON la cual es lanzada cuando detecta que se está produciendo un ataque de desasociación de un cliente de la red generando una denegación de servicio.La relación que existe entre una y otras, es que la vulnerabilidad encontrada en WPA, es explotada por medio de un ataque de desasociación (desasocia a los clientes de red del protocolo WPA) de un cliente o varios que se encuentren en la red y este es detectado por kismet, generando un evento en el SIEM e inmediatamente activa esta directiva y genera como respuesta

el reinicio del sistema con el de desconectar al atacante del punto de acceso y envía un correo dueño del sistema para informarle de la situación.

Para probar esta directiva, se tiene que primero se identifica la vulnerabilidad en un dispositivo IoT relacionada a el protocolo WPA y WPS2, con ayuda de la herramienta de Openvas se detecta esta vulnerabilidad del dispositivo, para luego dejar que Kismet detecte un ataque de desasociacion del dispositivo de la red.

Para el caso de Kismet, esta herramienta identifica este ataque con la alerta llamada "**BCASTDISCON",** que nos indica que se está realizando un ataque que aprovecha la vulnerabilidad de dispositivo representada en el CVE-2017-13077.

Una vez la herramienta OSSIM recibe los eventos generados por las herramientas Openasvas y Suricata IoT, este debido a su configuración de la directiva de correlación genera una respuesta ejecutado un script en el dispositivo centinela, que a su vez ejecuta un script que reinicia el dispositivo atacado.

El orden de la ejecución de esta directiva queda de esta manera:

1. En este escenario vamos a generar la alerte BCASTDISCON la cual se dispara cuando detecta que hay un ataque de desasociación de la red de un cliente o de varios, causando una posible denegación de servicio.

   Debemos tener el Kismet corriendo:
   Comando : Kismet
2. Para generar el ataque vamos a utilizar la herramienta aireplay-ng, en nuestro caso vamos a desconectar a todos los clientes conectados a la red:
   Escribimos el siguiente comando: - aireplay-ng –deauth 0 –a < BSSID> wlan1

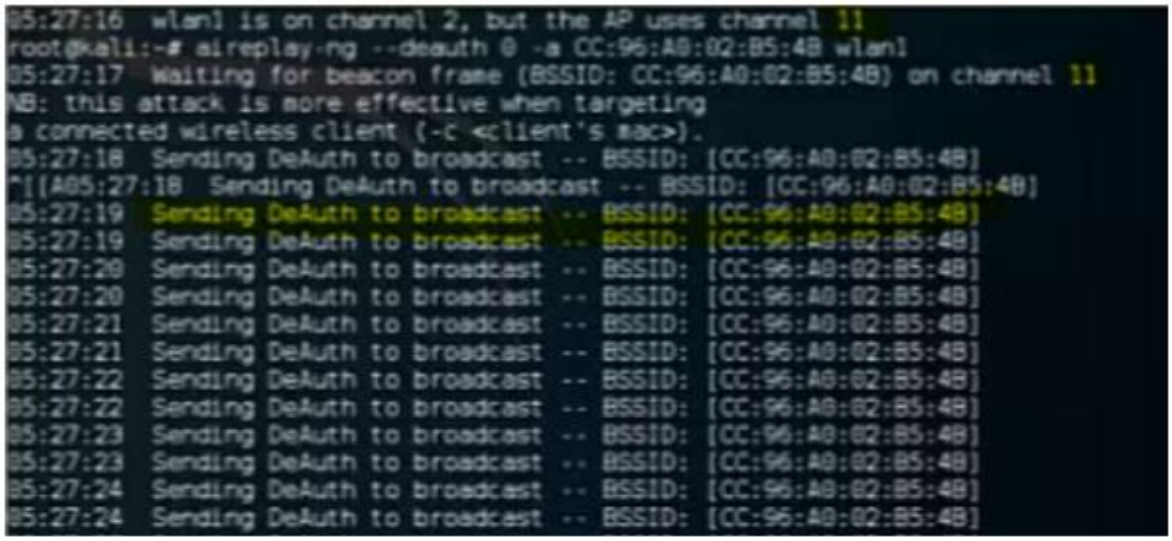


Ilustración 36, ataque WiFi sobre didpositivo IoT

3. Kismet detecta el ataque sobre el dispositivo.

Ilustración 37, alerta Kismet sobre ataque

4. OSSIM detecta el ataque sobre el dispositivo gracias al evento enviado desde kismet y la vulnerabilidad hallada por OpenVas sobre el Protocolo WAP:
   Las siguientes imágenes nuestran como llegan los eventos de esta regla de correlación al OSSIM:

Evento kismet

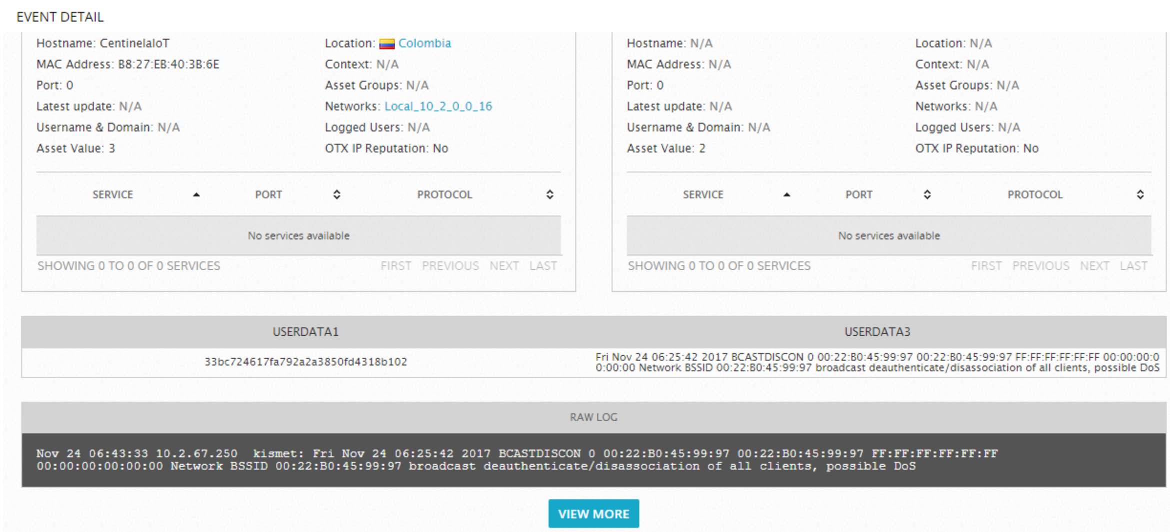


Ilustración 38, Evento Kismet directiva 3

Donde se muestra: el tipo de alerta que genero kismet, el punto de acceso del ataque y el tipo de ataque que se realizó.

Evento OpenVas

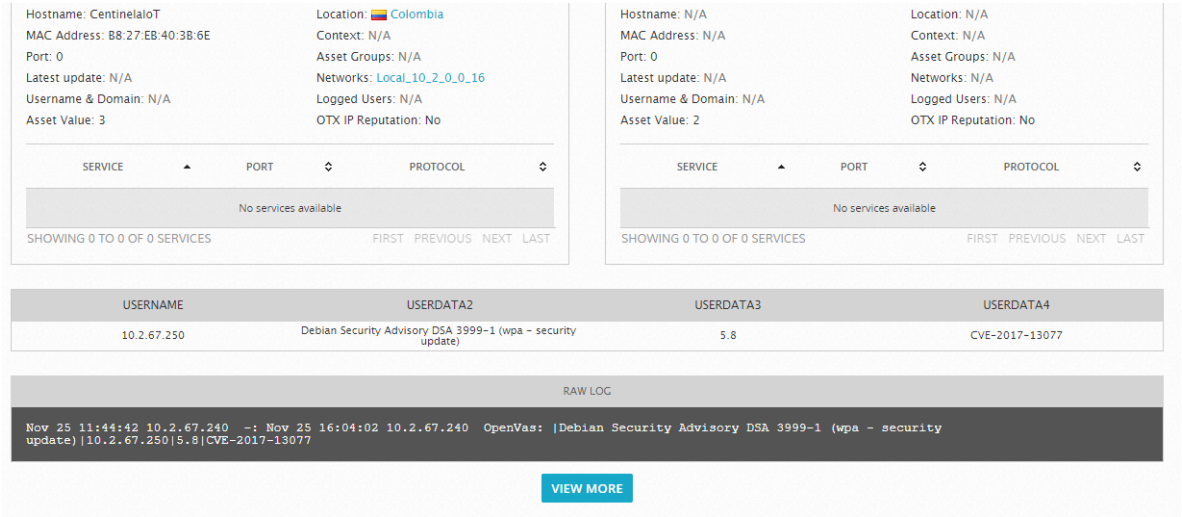


Ilustración 39 Evento Openvas directiva 3

En este evento se muestra la vulnerabilidad la cual está explotando el ataque que se está generando.

5. Gracias a la configuración de OSSIM se genera la correlación cruzada de eventos, esta consiste en tener diferentes fuentes de eventos de seguridad reportando que permiten inferir ataques de seguridad, en el caso de esta directiva de genera un evento de Openvas y otro evento de Kismet, lo cual al tener eventos de estas dos fuentes se activa la directiva de correlación cruzada. Cabe resaltar que para que se active esta directiva de correlación los eventos que llegan deben ser del mismo tipo, tanto la vulnerabilidad específica y el ataque.

Ilustración 40 Directiva de correlación cruzada 3

6. Se genera la respuesta activa, la cual es reiniciar el dispositivo IoT.

Para el caso de esta respuesta activa, cabe resaltar que se debe tener permisos de administrador del dispositivo que se va a realizar la operación de actualización en el dispositivo centinela para que esta respuesta funcione.

```
import paramiko
import sys
def sshCommand(hostname,port,username,password,command):
    sshClient = paramiko.SSHClient()

    sshClient.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    sshClient.load_system_host_keys()
    sshClient.connect(hostname,port,username,password)
    stdin, stdout, stderr = sshClient.exec_command(command)
    print(stdout.read())

if __name__=='__main__':
    sshCommand(sys.argv[0],22,'root','toor','reboot')
```

Ilustración 41, script de reiniciar dispositivo IoT

9. CONCLUSIONES

Dentro de la investigación se obtuvieron diferentes resultados, aunque todos son independientes, todos componen el núcleo de la solución planteada como lo es el uso de un dispositivo centinela IoT para asegurar los dispositivos IoT dentro de las arquitecturas de Smart Home, se obtuvieron en grandes rasgos 4 resultados de la investigación.

El primer resultado es el diseño e implementación de un dispositivo Centinela IoT basando su funcionamiento en el uso de tres herramientas de ciber defensa: Kismet, Openas y Suricata.

El segundo resultado es la generación y envío de eventos de seguridad asociados a un escenario de Smart Home hacia una plataforma de gestión de eventos SIEM (OSSIM) en la nube, generando la posibilidad del análisis de estos eventos para la correlación de los mismos.

El tercer resultado obtenido en el proyecto es la generación de directivas y reglas de correlación (3) en la plataforma SIEM (OSSIM) específicas para los dispositivos IoT presentes en las arquitecturas de Smart Home.

El cuarto resultado fue la generación de respuestas a los ataques vistos en el anterior punto, estas respuestas están catalogadas como respuestas activas en donde se identifica el ataque especifico y se plantea una respuesta sobre el dispositivo, generando una acción directa sobre él o la red, para mitigar el ataque sobre los diferentes dispositivos.

10. RECOMENDACIONES Y TRABAJOS FUTUROS

Debido a la adopción de diferentes soluciones IoT en ambientes del hogar, es imprescindible el adoptar una cultura de seguridad para los dispositivos IoT en arquitecturas de Smart Home, al tener diferentes dispositivos conectados a internet la cantidad de vulnerabilidades que trae consigo el uso de estos dispositivos aumenta en una cantidad muy grande, por lo tanto la solución planteada del uso del dispositivo centinela puede ser una aproximación para brindar una barrera en contra de los ataques que pueden sufrir los dispositivos IoT en las casas.

Se espera que para futuros trabajos se piense que mejorar el nivel de procesamiento del dispositivo IoT, lo cual permitiría tener una mejor gama de herramientas que permitan obtener un mayor número de eventos de seguridad de los dispositivos al alcance del centinela, e incluso se podría plantear tener una SIEM ligera que permita tener la correlación de eventos de manera local, lo cual evitaría tener una solución en la nube.

11. REFERENCIAS BIBLIOGRÁFICAS

[1] C. Stamford, “Gartner’s 2016 Hype Cycle for Emerging Technologies Identifies Three Key Trends That Organizations Must Track to Gain Competitive Advantage,” *Gartner*, 2016. .

[2] S. Li, L. Da Xu, and S. Zhao, “The internet of things: a survey,” *Springer Sci. Media*, pp. 243–259, 2015.

[3] A. Gantait, J. Patra, and A. Mukherjee, “Protegiendo dispositivos y gateways IoT,” 2016.

[4] C.-F. y D. E. en T. co. TeleSemana.com-Rafael A. Junquera, “Gartner proyecta inversiones en seguridad IoT, ¿son las cifras suficientes para frenar las amenazas? | TeleSemana.com.” .

[5] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, “Security and privacy challenges in industrial internet of things,” *Proc. 52nd Annu. Des. Autom. Conf. - DAC ’15*, vol. 17, pp. 1–6, 2015.

[6] V. A. B. B. Gupta, “Security in Internet of Things : issues , challenges , taxonomy , and architecture,” *Telecommun. Syst.*, 2017.

[7] International Telecommunication Union - ITU, “ITU Internet Reports 2005: The Internet of Things,” Genova, 2005.

[8] S. Li, T. Tryfonas, and H. Li, “The Internet of Things: a security point of view,” *Internet Res.*, vol. 26, no. 2, pp. 337–359, Apr. 2016.

[9] Y. Chahid, M. Benabdellah, and A. Azizi, “Internet of things security,” in *2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS)*, 2017, pp. 1–6.

[10] R. Van Rijswijk-Deij and E. Poll, “Using Trusted Execution Environments in Two-factor Authentication: comparing approaches.”

[11] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, and G. Vassilacopoulos, “Enabling data protection through PKI encryption in IoT m-Health devices,” in *2012 IEEE 12th International Conference on Bioinformatics & Bioengineering (BIBE)*, 2012, pp. 25–29.

[12] K. Fan, P. Song, and Y. Yang, “ULMAP: Ultralightweight NFC Mutual Authentication Protocol with Pseudonyms in the Tag for IoT in 5G,” *Mob. Inf. Syst.*, vol. 2017, pp. 1–7, Apr. 2017.

[13] W. Bae and J. Kwak, “Smart Card-Based Secure Authentication Protocol in Multi-Server IoT Environment,” *doi.org*, May 2017.

[14] GSM Association, “IoT Security Guidelines Endpoint Ecosystem Antitrust Notice,” 2016.

[15] D. S. Lavrova, “An approach to developing the SIEM system for the Internet of Things,” *Autom. Control Comput. Sci.*, vol. 50, no. 8, pp. 673–681, Dec. 2016.

[16] P. Zegzhda, D. Zegzhda, M. Kalinin, A. Pechenkin, A. Minin, and D. Lavrova, “Safe Integration of SIEM Systems with Internet of Things,” in *Proceedings of the 9th International Conference on Security of Information and Networks - SIN ’16*, 2016, pp. 81–87.

[17] I. Kotenko and A. Chechulin, “Computer attack modeling and security evaluation based on attack graphs,” in *Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems, IDAACS 2013*, 2013, vol. 2, pp. 614–619.

[18] I. Kotenko and A. Chechulin, “Common Framework for Attack Modeling and Security Evaluation in SIEM Systems,” in *2012 IEEE International Conference on Green Computing and Communications*, 2012, pp. 94–101.

[19] OWASP, “Top 10 2014-I6 Insecure Cloud Interface - OWASP.” .

[20] “OWASP Internet of Things Project - OWASP,” 2017. [Online]. Available: https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=IoT_Event_Logging_Project. [Accessed: 05-Apr-2017].

[21] D. O’Halloran and elena Kvochko, “Industrial Internet of Things: Unleashing the Potential of Connected Products and Services,” *World Econ. Forum*, 2015.

[22] T. DeCanio, “Suricata IoT - repository,” 2016. [Online]. Available: https://github.com/decanio/suricata-IoT/wiki/Suricata-IoT. [Accessed: 10-Dec-2017].

[23] “Top 10 2013-A1-Injection - OWASP, “Top 10 2013-A1-Injection - OWASP.” [Online]. Available: https://www.owasp.org/index.php/Top_10_2013-A1-Injection. [Accessed: 20- May-2017].

|

12. ANEXOS

12.1 TABLA DE HERRAMIENTAS OSSIM

Herramienta	Tipo (Activa/Pasiva)	Funcionalidades principales	Funcionalidades adicionales	Uso dentro de OSSIM	Posible utilidad para IoT	¿Existe una versión lite?
SNORT https://www.snort.org/	Pasiva	Análisis de tráfico en tiempo real generando eventos de seguridad. Combina inspecciones sobre firmas, protocolos y basadas en anormalidades. Motor de detección de ataques. Ofrece una serie de reglas y filtros pre configurados para el análisis (divididas en	Puede funcionar como Sniffer. Guarda archivos de logs sobre los análisis realizados.	IDS por defecto dentro de OSSIM (NIDS)	Detección de tráfico malicioso en la red por medio del análisis de payloads. Reglas pre configuradas con patrones específicos para ecosistemas IoT.	No, pero es posible compilarla e instalarla desde el código fuente en Raspbian Jessie

|

		cabecera y opciones).				
Suricata https://suricata-ids.org/	Pasiva	Sistema de detección de intrusos, IPS y motor de seguridad de red de alto rendimiento. Puede realizar análisis multihilo y le podemos asignar dependiendo de los recursos. Análisis offline de archivos PCAP. Al igual que Snort se basa en reglas para realizar los análisis.	Permite mostrar estadísticas de rendimiento. Detección de protocolos automáticos.	IDS adicional dentro de OSSIM (NIDS)	Podría ser un buen NIDS, además de tener una versión para IoT, pero al ser multihilo hay que revisar la capacidad de procesamiento	Tiene una versión especial para protocolos, plataformas y dispositivos IoT
PRADS PRADS. Passive Real-time	Pasiva	Detección de activos basado en firmas. Manejo de inventarios	Correlación de inventarios Detección de violación de políticas	Detección de Activos de forma pasiva	Se podría utilizar para realizar el mapeo del ecosistema IoT en tiempo real.	No, pero está disponible para Raspbian Jessie

|

Asset Detection System. https://github.com/gamelinux/prads		Información sobre activos y disponibilidad. Mapeo de activos con descubrimiento de servicios activos.				
NMAP Network Mapper https://nmap.org/	Activa	Escáner de seguridad para descubrimiento de hosts y servicios en la red. Inventariado de la red.	Herramienta para monitoreo de disponibilidad de servicios y hosts Escaneo de vulnerabilidades	Detección de activos y mapeo de forma activa y administración de resultados	Escaneo del ecosistema IoT obteniendo información de host y servicios y monitoreo de disponibilidad.	Está disponible para Raspbian Wheezy.
OCS Inventory NG https://www.ocsinventory-ng.org	Activa	Administración de activos y solución de despliegue. Mapea los activos con su composición a nivel de hardware y software. Descubrimiento de activos en la red.	Ip Discover Sus agentes son de bajo consumo de recursos. Monitoreo de vulnerabilidades Información de violación de políticas	Detección de activos. Inventario de software y hardware.	Mapeo de activos	Está disponible para dispositivos Android.

|

		Despliega agentes para la administración de activos.				
Openvas http://www.openvas.org/	Activa	Herramienta para escaneo de vulnerabilidades autenticada y no autenticada. Escaneo de vulnerabilidades sobre la red.	Permite establecer niveles de agresividad y de profundidad. Escaneo concurrente.	Escaneo de vulnerabilidades agendados y repetitivos. Reporte de vulnerabilidades sobre activos de la red.	Escaneo de vulnerabilidades sobre ecosistemas IoT. En algunos foros se habla de que la aplicación falla al escanear activamente en estos ecosistemas.	No, pero está disponible en la versión lite de Kali Linux.
Nikto https://cirt.net/Nikto2	Activa	Escaneo de vulnerabilidades sobre servidores web	Captura de cookies/prints. Escaneo de múltiples Host/puertos. Revisión de servidores desactualizados.	Detección de: Configuraciones erróneas de servidor y software. Archivos y programas por defecto. Archivos y programas inseguros. Software desactualizado.	Podría ser una buena herramienta para el escaneo de servidores además de brindar funcionalidades de revisión de estado del activo.	No
OSSEC https://ossec.github.io/	Activa	HIDS – Host-based intrusion detection.	FIM – Monitor de integridad de archivos.	Administración de despliegues distribuidos.	Se podría utilizar en forma de servidor para la correlación de	No, pero puede ser instalado en diversas versiones de Windows y Linux.

|

		Motor de correlación de eventos. Análisis de logs. Detección de rootkits Sistema de alertas y respuestas en tiempo real.	Monitoreo de registro de Windows. Aplicación centralizada de políticas.	Envío de eventos a través de agentes. Herramienta de monitoreo de integridad de archivos y registro.	eventos y análisis de logs. Envío de eventos de dispositvos IoT. (Pero hay que configurarlo para que no gaste tantos recursos)	
KISMET https://www.kismetwireless.net/	Pasiva	IDS para redes inalámbricas 802.11. Detección de redes por medio de la recolección pasiva de paquetes. Sniffer.	Soporta plugins para redes que no sean 802.11 como bluetooth, DECT y otros. Penetration testing.	Monitoreo de redes WIFI. Detección de Rogue apps	Puede ser utilizado para el monitoreo de los diferentes dispositivos IoT presentes en redes inalámbricas.	Está disponible para Raspbian.
NAGIOS https://www.nagios.org/	Activa	Observación de hosts y servicios generando alertas. Monitoreo de activos. Visualización de información sobre los datos obtenidos.	Administración de activos	Monitoreo de disponibilidad. Visibilidad de puertos abiertos o cerrados.	Monitoreo de activos presentes en el ecosistema y de los servicios que proveen. Chequeos programados.	Sí, Nagios Core que tiene los servicios básicos de monitoreo.

|

		Detección de actividades maliciosas.				
TCPDUMP http://www.tcpdump.org/	Activa	Analizador de paquetes por línea de comandos y libpcap. Muestra información de contenidos de los paquetes capturados, puede guardar capturas y analizarlas más tarde.	Herramienta para ataques de MIM	Revisión de hosts y envío de alertas. Monitoreo de hosts y dispositivos remotamente, por medio de hosts y plugins.	Monitoreo del comportamiento de los dispositivos	Sí, es una herramienta de bajo consumo de recursos.
FProbe http://fprobe.sourceforge.net/	Pasiva	Herramienta Libcap based que recolecta datos de tráfico de la red.	Es utilizada para realizar port mirroring en dispositivos que no tienen capacidad de flujo.	Ver flujos de información para asistir la respuesta de incidentes.	Captura de paquetes y monitoreo sin ser invasiva.	No, pero está disponible para Raspian Jessie.
NFDump http://nfdump.sourceforge.net/	Pasiva	Lee flujos de datos de archivos almacenados por nfcapd y procesa	Análisis de flujos de TCPDUMP	Análsis de flujos de redes. Generación de estadísticas.	Podría servir como herramienta de análisis, pero siento	No, pero está disponible para Raspian Jessie.

|

		los flujos de acuerdo a las opciones dadas.			que no será muy provechosa	
NfSen - Netflow Sensor http://nfsen.sourceforge.net/1.2.4/index.html	Pasiva	Interfaz web para la herramienta NFDump. Muestra flujo de datos, paquetes, y bytes.	Escribir plugins para NFDump.	Mostrar datos de los flujos de red. Procesamiento de datos con un intervalo de tiempo específico.	Interfaz web de bajo consumo para análisis de flujo de datos.	No, pero está disponible para Raspian Jessie.
NTop http://www.ntop.org/	Pasiva	Provee datos de uso de red en tiempo real e históricos	Realizar predicciones de comportamiento futuro.	Con las predicciones de flujo de datos, genera alertas si difiere. Estadísticas de uso de red. Información de activos.	Me parece una buena herramienta para el análisis de uso de la red, además su funcionalidad de predicción con el algoritmo RRD Aberrant Behavior podría ser usado para generar alertas.	Es usada para el monitoreo de ecosistemas IoT.

12.2 **AUTORIZACION DE PUBLICACIÓN DE DOCUMENTOS EN EL REPOSITORIO COLECCIONES DIGITALES DE LA ESCUELA COLOMBIANA DE INGENIERIA JULIO GARAVITO**

AUTORIZACIÓN DE PUBLICACIÓN DE DOCUMENTOS EN EL REPOSITORIO COLECCIONES DIGITALES DE LA ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

Fecha: 19-01-2018

1. **Datos de la publicación (trabajo de grado, artículo, video, conferencia, libro, imagen, fotografía, audio, presentación u otros) y del (los) autor(es)**

Documento de Identidad		Apellidos	Nombres	Correo Electrónico
Tipo	Número			
CC	1020795099	Moreno Jurdiaga	Nicolas	nicomorenoju@gmail.com
CC	1020819816	Vega Torres	Andres Felipe	andres.vega52@hotmail.com
CC	1072658469	Morón Castro	Stefany Jhoana	stefamoron25@gmail.com

Título del Documento	Seguridad para IOT: Solución para la gestión de eventos de seguridad de Arquitectura de Internet de las cosas.
Nombre del evento origen (si aplica)	
fecha del evento (si aplica)	
Palabras clave	SIEM, eventos, ataque, IOT, directivas de correlación.

Acuerdos de Confidencialidad: **No tiene Acuerdo(s)** ☒ **Tiene Acuerdo(s)** ☐
(Si tiene acuerdos de confidencialidad, por favor diligencie el siguiente cuadro)

Persona Jurídica o Natural	Desde			Hasta		
	DD	MM	AA	DD	MM	AA

2. Autorización de publicación de documentos en el Repositorio Institucional

Autorizo a la Escuela Colombiana de Ingeniería Julio Garavito para publicar el trabajo de grado, artículo, video, conferencia, libro, imagen, fotografía, audio, presentación u otro (en adelante documento) que en la fecha entrego en formato digital, y le permito de forma indefinida que lo publique en el repositorio institucional, en los términos establecidos en la Ley 23 de 1982, la Ley 44 de 1993, y demás leyes y jurisprudencia vigente al respecto, para fines educativos y no lucrativos. Esta autorización es válida para las facultades y derechos de uso sobre la obra en formato digital, electrónico, virtual; y para usos en redes, internet, extranet, y cualquier formato o medio conocido o por conocer. En mi calidad de autor, expreso que el documento objeto de la presente autorización es original y lo elaboré sin quebrantar ni suplantar los derechos de autor de terceros. Por lo tanto, es de mi exclusiva autoría y, en consecuencia, tengo la titularidad sobre él. En caso de queja o acción por parte de un tercero referente a los derechos de autor sobre el documento en cuestión, asumiré la responsabilidad total y saldré en defensa de los derechos aquí autorizados. Esto significa que, para todos los efectos, la Escuela actúa como un tercero de buena fe. Toda persona que consulte el Repositorio Institucional de la Escuela, el Catálogo en línea u otro medio electrónico, podrá copiar apartes del texto, con el compromiso de citar siempre la fuente, la cual incluye el título del trabajo y el autor. Esta autorización no implica renuncia a la facultad que tengo de publicar total o parcialmente la obra en otros medios.

Esta autorización está respaldada por las firmas del (los) autor(es) del documento.

Sí autorizo (amos)

3. Firmas de autor(es)

Firma autor 1

Documento de identidad:
1020796088

Firma autor 2

Documento de identidad:
1020798816

Firma autor 3

Documento de identidad:
1072658961

4. Firmas de aprobación

Director del Trabajo de Grado

Daniel Díaz

Documento de identidad: 1061695713

Director del Programa (Si aplica)

Documento de identidad: