A. Acien, A. Nieto, and J. Lopez, "Modelo para la clasificación y análisis de ataques Cross-Platform", *IV Jornadas Nacionales de Investigación en Ciberseguridad (JNIC 2018)*, 2018.
NICS Lab. Publications: https://www.nics.uma.es/publications

Modelo para la clasificación y el análisis de ataques *cross-platform*

Antonio Acien, Ana Nieto y Javier Lopez
Network, Information and Computer Security (NICS) Lab
Lenguajes y Ciencias de la Computación
Universidad de Málaga, España
Email:{acien,nieto,jlm}@lcc.uma.es

***Resumen*—Los ataques *cross-platform* suponen un serio desafío para los mecanismos de seguridad cuando los portadores de un ataque dirigido no son conscientes de su participación en el mismo. Es por ello que, con dispositivos y tecnologías cada vez más entrelazadas, en constante comunicación, numerosos ataques pasan desapercibidos hasta que alcanzan su objetivo final. Estos nuevos escenarios hacen posible una vía de transmisión a tener en cuenta, y que se debe abordar cuanto antes, ya que sus consecuencias, especialmente en el panorama de telecomunicaciones actual, podrían ser desoladoras. La rápida transmisión de estos ataques, y la dificultad que supone su prevención, detección y mitigación antes de que se hagan efectivos, hacen que el problema sea particularmente preocupante. En este artículo se presentará un modelo para el análisis de los ataques cross-platform silenciosos, cuyo objetivo es ayudar a comprender mejor este tipo de amenazas y ofrecer soluciones que permitan mitigarlas y rastrearlas.**

***Index Terms*—Cross-platform, architecture, attack, security.**

I. Introducción

Los ataques *cross-platform* son aquellos que afectan a múltiples plataformas y servicios, difíciles de detectar y rastrear, ya que la mayoría de medidas de seguridad se restringen a la seguridad de una plataforma. En este artículo, nos centramos en ataques cross-platform en los que el funcionamiento de los portadores intermedios no se ve directamente alterado y por ello son difíciles de detectar hasta que alcanzan su objetivo final. En un entorno en el que las distintas plataformas, tecnologías y protocolos están cada vez más interconectadas entre sí, este tipo de ataques representan un serio riesgo.

A lo largo de este artículo se verá cómo una plataforma puede ser comprometida desde otra muy distinta, empleando elementos intermediarios en la comunicación para pivotar. Dado que, a nuestro juicio, no hay modelos para el análisis de este tipo de ataques que permita entender su progresión en tiempo real, en este artículo se propone un modelo para el análisis de ataques cross-platform, con el objetivo de ayudar a su detección, prevención y mitigación.

Cabe destacar que la intercomunicación de distintas plataformas, o distintas capas dentro de una misma (p.ej. el caso de los *network slices* en 5G), es un escenario cada vez más recurrente en las comunicaciones modernas. No es raro tener un smartphone que se conecte a un ordenador, y éste a su vez a varios dispositivos más (routers, *gadgets* USB, o incluso otros smartphones), o incluso a un coche que también cuenta con una conexión a una red inalámbrica. También se halla un panorama similar en entornos virtualizados (conectándose el huésped al anfitrión) o en sistemas compuestos, a su vez, de varios sistemas empotrados que se comunican. La idea de un ataque infectando una de estas plataformas y propagando su actuación a las que se comunican con ella es cada vez más una realidad, que no sólo se ha dado en pruebas de concepto o despliegues controlados, sino en entornos de producción, y a usuarios medios (sección II). Con la aparición de nuevas tecnologías que fomentan estas interacciones (5G, mmWave, SDN...), es necesario establecer herramientas de análisis que entiendan este cambio de contexto.

El artículo se estructura como sigue. En la sección II se presenta un estudio del estado del arte en lo referente a ataques Cross-platform, y tras ello se presentará el modelo propuesto, llamado BTV (*Bearer, Transmitter, Victim*), en la sección III, que se formalizará en la sección IV. La sección V describe cómo se aplicaría el modelo propuesto en un caso de uso simplificado. Por último, se lleva a cabo una discusión de los resultados y unas conclusiones derivadas del trabajo realizado en las secciones VI y VII, respectivamente.

II. Análisis del Estado del arte

La definición de Cross-platform es relativamente amplia, pudiendo referirse a varias interpretaciones. Desde ataques que, con el mismo código, pueden afectar a varias plataformas (por ejemplo, aprovechando la naturaleza multi-plataforma de Java [1]), hasta aquellos que modifican su código dependiendo del sistema al que se dirijan (ataques metamórficos) [2]. Además, algunos trabajos sobre Cross-platform optan por centrarse en permisos y políticas para escenarios específicos, o procedimientos, métodos y librerías que tienen partes correspondientes en diferentes sistemas operativos o arquitecturas [3].

La mayoría de las contribuciones en este área se ha enfocado en redes en general: transmisión de paquetes, autenticación y autorización (control de acceso) entre otros temas [4]. Además, las redes que se han tenido en cuenta han sido casos concretos, como 3G [5] [6]. También se estudian casos muy específicos como en redes vehiculares, donde los vehículos son plataformas que integran otras plataformas. Por ejemplo, en [7] se emplea un CD preparado para causar alteraciones en el motor o la dirección, y en [8] un dispositivo vinculado envía mensajes a otras partes del vehículo, como los nodos

multimedia o los encargados de gestionar las comunicaciones externas.

También se han dado casos en escenarios de virtualización, donde el equipo anfitrión (host) es una plataforma que se sobre-entendía aislada del equipo virtualizado. Sin embargo, de sobra es sabida la existencia de ataques diseñados para escapar la barrera invitado-anfitrión, permitiendo que desde la máquina virtual se pueda escribir en zonas de memoria reservadas para la máquina que la hospeda. Estos ataques van más allá de las pruebas de concepto[9], habiendo exploits disponibles comercialmente para llevar a cabo el ataque en un entorno real [10].

El *Internet de las Cosas* (IoT) tampoco es una excepción. Por ejemplo, la versión para Windows de Mirai ataca en primera instancia a ordenadores con este sistema operativo, pero una vez llega a ellos, escanea su red local en búsqueda de dispositivos IoT vulnerables, los cuales pueda hacer parte de una botnet para llevar a cabo ataques de denegación de servicio distribuidos hacia otras infraestructuras [11].

Asimismo, se han documentado ataques pasando de smartphone a ordenador, y viceversa. En el caso de los primeros, normalmente se transmiten cuando el dispositivo móvil se conecta por USB, pinchando el micrófono del ordenador y enviando el audio grabado a un servidor externo, espiando conversaciones [12]. Siguiendo el flujo contrario, se encuentran ataques que se pasan del ordenador a un dispositivo Android, suplantando las apps que tiene instaladas por versiones aparentemente idénticas, que envían los datos de identificación introducidos a un servidor del atacante [13]. Esto es especialmente delicado cuando se trata, por ejemplo, de datos bancarios. También cabe destacar el caso de XcodeGhost, una versión modificada del entorno de desarrollo para iOS que infecta las aplicaciones elaboradas con el mismo y transmite este malware a los dispositivos donde se instalen [14].

Cabe destacar que los ataques Cross-plaform, aunque más preocupantes y comunes en los últimos tiempos, principalmente motivados por las amenazas persistentes avanzadas, no son nada nuevo, aunque la terminología y el enfoque que se le da en este artículo, más general, sí lo sea. La propagación de ataques a través de múltiples plataformas es algo que ocurre desde varias generaciones de comunicaciones atrás, y queda patente que aunque se lleven a cabo mejoras en la seguridad, sin soluciones que se habitúen a los cambios de contexto será muy difícil frenar la propagación de los ataques en los elementos portadores. Cualquier modelo de seguridad debería considerar este tipo de ataques y afrontarlos desde una perspectiva global. A medida que se desarrollan nuevas plataformas y tecnologías, también se desarrolla malware específico para ellas [15], lo que pone de manifiesto la urgencia de afrontar este problema. Por lo tanto, definir una estrategia y medidas frente a estos ataques es prioritario.

Este artículo proporcionará un enfoque general a este problema con el objetivo de ofrecer soluciones que permitan dar respuesta con independencia de la plataforma.

III. Modelo para el análisis de ataques Cross-platform

En esta sección se presenta el modelo BTV (*Bearer, Transmitter, Victim*). Su objetivo es cubrir los escenarios de ataques Cross-platform de una manera simple y general, con flexibilidad para adaptarse a cada uno de los diferentes escenarios y flujos de ataque. La idea principal detrás del concepto de BTV es la de proponer un esquema simple que consiste en un agente que alberga el ataque, conocido como *portador o bearer*. Este ataque se dirige a una víctima, a la cual no infecta directamente. En su lugar, el ataque pasa a través de otro agente, llamado *transmisor o transmitter*, que contiene el ataque, pero no ve su funcionamiento alterado (al menos no hasta el punto de poder detectar el ataque). Por último, el transmisor enviará inconscientemente el ataque a la víctima, que será quien sufrirá las consecuencias del mismo. Tras mostrar esta idea de una forma gráfica y detallada (sección III-A), se explicará cómo se emplearía el modelo para identificar ataques crosss-platform (sección III-B).

III-A. Componentes

El enfoque clásico llevado a cabo cuando se analizan ataques (sección II) es un escenario simplificado de dos actores. En dicho escenario, tanto el atacante como la víctima son agentes bien definidos: el atacante se esfuerza en propagar la infección a la víctima, la cual se ve comprometida y dañada. Sin embargo, este punto de vista puede quedarse corto en los ataques Cross-platform, los cuales involucran más sistemas con roles diferentes. Los sistemas normalmente están preparados para detectar el malware que se dirige a ellos, pero no a otros. Cuando surge una situación en la que un dispositivo se infecta pero su funcionamiento no cambia, detectar tal infección y evitar su propagación no es nada trivial. El rol de este transmisor inconsciente es esencial en muchos casos, porque permite a los ataques afectar a dispositivos a los que, de otro modo, no tendrían acceso.

El modelo BTV (Figura 1) permite estudiar la posibilidad de un ataque originado en una plataforma extendiéndose a otras distintas, aplicándose desde escenarios novedosos como 5G y SDN, o vehículos con conexiones de datos, hasta situaciones más cotidianas como conexiones USB o Bluetooth entre smartphones y ordenadores.

En el esquema hay tres actores principales, que se describen a continuación:

1. **Portador:** este participante en el esquema es el que comienza la propagación del ataque. Puede ser un atacante malicioso, con el objetivo de causar daño, o un usuario inconsciente que cree estar operando con normalidad.
2. **Transmisor:** es el agente que recibe el ataque, pero no se ve afectado directamente por él.
3. **Víctima:** el objetivo del ataque. Normalmente se comunica con un servidor CnC (*Command and Control*) para recibir instrucciones o al que enviar datos, aunque no siempre es el caso, ya que podría ser víctima de un ataque que no necesite comunicación externa.

Como se aprecia en Figura 1, hay más partes involucradas. Una de ellas es la tecnología de acceso, la cual, dependiendo

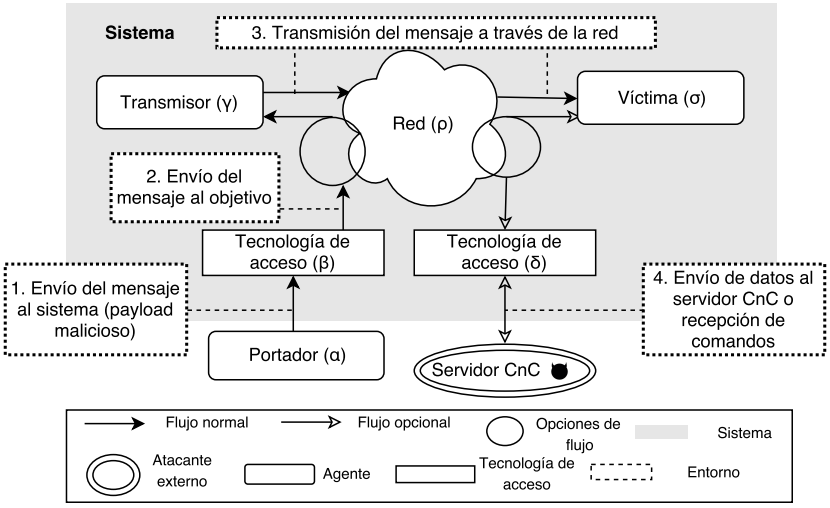

Figura 1. Diagrama del modelo BTV

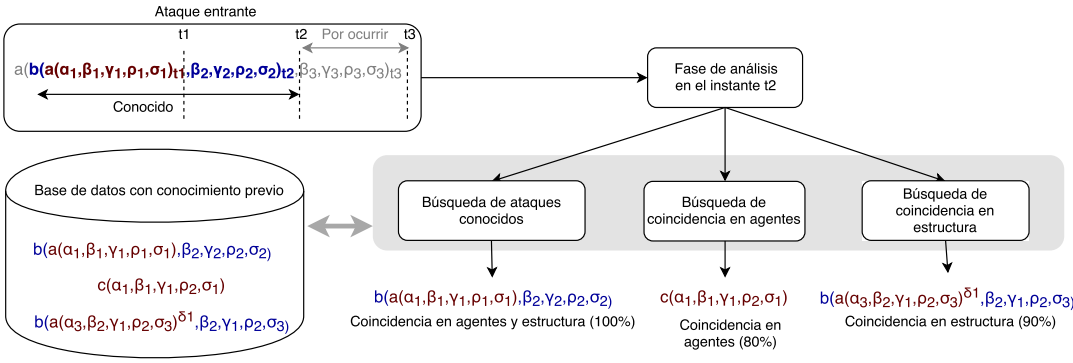

Figura 2. Implementación de un sistema de detección de amenazas basado en el modelo BTV

del tipo de ataque, variará. Si el portador es un servidor de Internet con malware, podría acceder al transmisor simplemente a través de un router normal. Si en cambio, se trata de un vehículo, este acceso será una puerta de enlace para comunicaciones externas o una ECU (*Electronic Control Unit*, unidad de control electrónica) en un sistema SCADA (*Supervisory Control and Data Acquisition*, control de supervisión y adquisición de datos).

También presenta una red que expresa la conexión entre el transmisor y el agente afectado, sea via WiFi, USB, Bluetooth, CAN o cualquier otro protocolo. Los mensajes pueden ser enviados a través de ella a cada uno de los agentes, o directamente a través de la tecnología de acceso.

Cabe destacar que, mientras que en algunos escenarios los agentes son dispositivos independientes que tienen conexiones momentáneas (como un smartphone y un ordenador), en otros son parte de un sistema completo e indivisible, donde comparten recursos o mensajes, tales como las distintas partes de un vehículo, o una máquina virtual y su anfitrión. Además de estos sistemas, se pueden definir entornos, siendo estos escenarios donde normalmente varios agentes interactúan entre ellos.

III-B. Despliegue

Al tener la posibilidad de analizar los ataques de manera sistemática mediante este modelo, surgen aplicaciones de la misma que, mediante una implementación, ayudan a prevenir, detectar y mitigar estas amenazas.

Una vez analizados los ataques Cross-platform conocidos, ya sean aquellos presentes en bibliografía, o los que se detecten en entornos de producción y sean analizados, se pueden usar sus patrones de propagación como conocimiento previo. Si se incluyen en una base de datos, comparar sus participantes o su estructura con las amenazas detectadas, puede ayudar a predecir su evolución, y prevenir su propagación o tomar las medidas necesarias para mitigarlas.

La comparación con los ataques conocidos se puede realizar de varias maneras. Se puede comparar de manera estructural, viendo así si los agentes implicados siguen un cierto patrón y predecir si el ataque se extenderá a otra plataforma. También se puede comprobar si los agentes implicados coinciden con los de algún caso conocido, sabiéndose así si se trata de algún ataque visto previamente, o quizás una variación del mismo.

Al llevar a cabo estas comprobaciones con los ataques de la base de datos, se ha de establecer un criterio que permita indicar el índice de similitud. Esto ayudará a decidir qué

medidas tomar, ya que si el parecido a una cierta amenaza es alto, servirá de referencia para hacer frente a la actual.

Este criterio de similitud se puede basar en distintas observaciones. Una de las comparaciones a realizar es estructural: si el ataque sigue la misma estructura que alguno conocido, aunque mediante distintos agentes, se puede predecir si se va a producir una propagación del mismo y cómo. En cambio, si la estructura es distinta, pero coinciden los agentes, se puede prever a qué agentes más podría afectar si sigue extendiéndose con dicha estructura. Por supuesto, pueden coincidir ambas situaciones, y corresponderse totalmente el ataque con uno ya conocido. Estos casos se ilustran en la Figura 2. Cabe destacar, que las flechas que marcan la correspondencia de los ataques con otros vectores similares de la base de datos cambiarían conforme se tenga más información del ataque, expresado a través de la barra de tiempo t.

Como ya se ha indicado, una de las mayores dificultades que presentan los ataques Cross-platform es su detección, ya que las plataformas normalmente están preparadas para detectar y mitigar ataques dirigidos específicamente a ellas, y esta tarea se dificulta si para ellas el ataque es inocuo y sólo colaboran en su expansión. Mediante este conocimiento previo expresado con el modelo se lograría hacer frente a este problema, ya que al detectar un ataque y compararlo con los datos, se conocerían las posibilidades de expansión que posee y el posible funcionamiento de la misma.

IV. Formalización

Con objetivo de ayudar a la definición del modelo, y hacer que tenga una aplicación más directa y comprensible para una máquina, se ha decidido formalizar su estructura con un lenguaje sencillo pero versátil, de manera que se pueda instanciar en cada uno de los casos necesarios. Al hacer que se pueda expresar de una manera uniforme, también se posibilita que sea fácilmente comprensible por máquinas, ayudando a su procesado automático. Esto ayudará a llevar a cabo la detección que se detalla en la sección V, ya que las comparaciones comentadas se pueden hacer de forma sistemática comparando variables o expresiones.

IV-A. Vector de ataque

La expresión general por la que se define el vector de ataque (V) es la siguiente:

$$V = f\,x\,{}_t^{\delta}\,,\ t \in \mathbb{N} \tag{1}$$

donde $f\,x$ es la función de flujo (sección IV-D), $delta$ es la tecnología de acceso (entre la víctima y el servidor CnC si lo hubiera, sección IV-C), y t es el instante de tiempo, que nos permite definir momentos para observar cómo profundiza o avanza el ataque. Las variables del vector x se detallan en la sección IV-C.

Mediante este lenguaje, que se sirve de distintas variables se puede cubrir cada caso contemplado por el modelo BTV. Esta expresión describirá los *vectores de ataque* (V) del modelo BTV. A continuación describimos de forma detallada los conjuntos, variables y funciones de flujo y delimitadores que proporcionan significado a esta fórmula.

IV-B. Conjuntos

Se definen dos conjuntos, A y T, para expresar los posibles agentes y tecnologías, respectivamente. Son conjuntos compuestos por cadenas de caracteres. Dado que no es posible incluir todas las posibilidades en estos conjuntos, se instanciarán en cada caso de uso, incluyendo los miembros necesarios.

Ejemplo:

$$A = \{Servidor, Smartphone, NodoCAN...\} \tag{2}$$

$$T = \{USB, WiFi, Bluetooth, ZigBee, CAN...\} \tag{3}$$

IV-C. Variables del modelo BTV

En el modelo se emplean varias variables, que se pueden ver de forma gráfica en la Figura 1. Aquí procedemos a describirlas.

$$\alpha, \gamma, \sigma \in \{V, A\} \tag{4}$$

$$\beta, \rho, \delta \in T \tag{5}$$

$$t \in \mathbb{N} \tag{6}$$

- α**:** Portador del ataque
- β**:** Tecnología de acceso al transmisor
- γ**:** Transmisor del ataque
- ρ**:** Red que comunica al transmisor con la víctima
- σ**:** Víctima del ataque
- δ**:** Tecnología de comunicación con el servidor CnC (si hay)

Como se aprecia en la exprs. (4), α puede ser a su vez otro vector de ataque, aportando así recursividad. Lo mismo sucede con el transmisor y la víctima del ataque. El instante de tiempo en el que se produce el ataque es representado por medio de t, pudiendo variar a medida que se conoce más información sobre el mismo. Por último, señalar que δ es opcional, ya que no en todos los escenarios se establece comunicación con servidores CnC.

IV-D. Funciones de flujo

Para representar el flujo del ataque en el vector, se pueden indicar cada uno de los componentes del modelo por los que pasa el ataque en su camino de manera secuencial. Sin embargo, esto haría que los vectores tuvieran un número de miembros distinto dependiendo de la situación, y que en varios de ellos se repitiera. Para evitar esta repetición, la función de flujo representa el camino del ataque por los distintos agentes a los que afecta con un solo símbolo. La función de flujo se expresa de forma general como f, pero se instanciará a funciones de flujo específicas que describen el flujo específico del ataque (véase la Figura 3): a, b, c o d. Estas funciones se usarán extensivamente en el caso de uso (Sección V), para expresar de manera simple el flujo de los ataques mostrados.

$$f\,x = \begin{cases} a\,x & \text{si}\, x = \alpha \to \beta \to \gamma \to \rho \to \sigma \\ b\,x & \text{si}\, x = \alpha \to \beta \to \boldsymbol{\rho} \to \gamma \to \rho \to \sigma \\ c\,x & \text{si}\, x = \alpha \to \beta \to \gamma \to \rho \to \sigma \to \boldsymbol{\rho} \\ d\,x & \text{si}\, x = \alpha \to \beta \to \boldsymbol{\rho} \to \gamma \to \rho \to \sigma \to \boldsymbol{\rho} \end{cases} \tag{7}$$

Hay que destacar que las funciones a y c son muy similares, y el único cambio que presentan es que, en caso de ocurrir una propagación del ataque (ya sea a otro entorno, o a un servidor de CnC), ésta se hará a través de la red. Por lo tanto, que un ataque *mute* de una función a otro en su funcionamiento es muy posible, ocurriendo lo mismo entre las funciones b y d.

IV-E. Delimitadores

Además de las variables previamente indicadas, también se incluyen los símbolos $[\,]$ y $\{\,\}$. Éstos se usan para delimitar la extensión de un sistema y un entorno, respectivamente. Tener unos agentes comprendidos entre los límites de un sistema significa que son indivisibles y tienen un funcionamiento conjunto. Por otra parte, los entornos representan conjuntos de dispositivos que normalmente actúan entre ellos para llevar a cabo ciertas funciones, pero pueden operar de manera independiente. Hay ciertas reglas a seguir a la hora de usar estos símbolos:

1. Tanto un entorno como un sistema pueden existir el uno sin el otro. Sin embargo, un entorno no puede estar comprendido dentro de un sistema. Un sistema sí puede estar comprendido dentro de un entorno.
2. En caso de que haya un sistema cuyo límite comience dentro de un entorno, su fin también tiene que estar dentro del entorno.
3. Tanto los delimitadores de sistema como los de entorno actúan sobre las variables $\alpha, \beta, \gamma, \rho$ y ϵ, no pudiendo aplicarse fuera de ellas.

Si, por ejemplo, hay un entorno que comprende la tecnología de acceso, el transmisor, y la red, y un sistema conformado por estos dos últimos, se expresará como $f(\alpha, \{\beta, [\gamma, \rho]\}, \sigma)$.

Una vez definido el contexto a analizar empleando esta formalización, llevar a cabo la implementación de un sistema de mitigación de ataques basado en el modelo BTV es más sencillo, ya que todos los ataques se pueden expresar mediante fórmulas con una sintaxis común.

V. Caso de uso

En esta sección se estudiará un caso de uso que implica varias tecnologías, observando así cómo el modelo BTV se puede aplicar a diferentes escenarios y protocolos, y mostrando su utilidad práctica en situaciones reales. Aunque en el caso de uso se mencionarán tecnologías y dispositivos específicos para explicarlo, cabe puntualizar que habría infinitas variaciones. Asimismo, se formalizará, poniendo en práctica las funciones y variables vistas en la sección anterior aplicadas tanto a dispositivos como tecnologías concretas.

V-A. Primera fase

En la primera fase del ataque, un servidor de internet que alberga malware infecta un ordenador a través de un *exploit* que se aprovecha de una vulnerabilidad en el navegador para instalar un archivo ejecutable con código malicioso (pasos 1 y 2 de Figura4). El usuario del ordenador no es consciente de que el malware se instala en su máquina porque no afecta

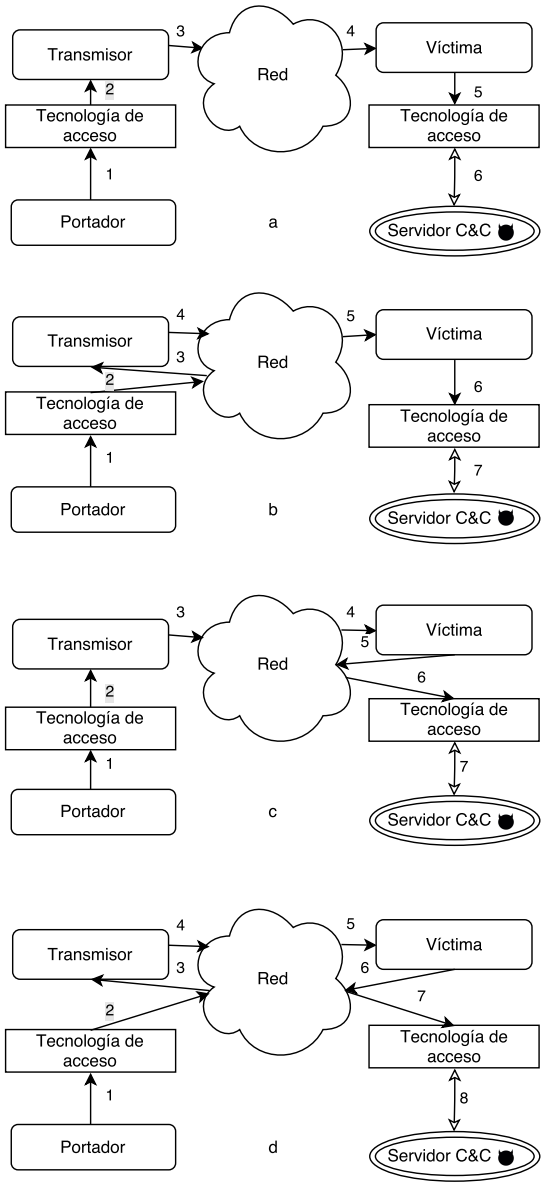

Figura 3. Posibles flujos del ataque que representan las distintas funciones

a su actividad normal, pero éste se encuentra en segundo plano monitorizando las conexiones USB hasta encontrar un terminal Android (paso 3). Cuando esto ocurra, el malware buscará una app que se conecte a un vehículo por Bluetooth, desinstalándola y sustituyéndola por una versión idéntica a ojos del usuario (paso 4), pero que se conecta con el servidor CnC del atacante (paso 5) y tiene propósitos maliciosos. Hay casos de malware que siguen este patrón de operación con otro tipo de aplicaciones, como por ejemplo, bancarias.

Así, el portador en esta fase sería el servidor externo, mientras que el ordenador y el dispositivo Android tendrían los roles de portador y víctima, respectivamente. A su vez, el smartphone podría ser portador si la cadena del ataque continúa.

Para expresar esta parte del caso de uso mediante la formalización presentada en la sección IV, se usarán las abreviaturas presentadas en la tabla I, para no extender la longitud de la

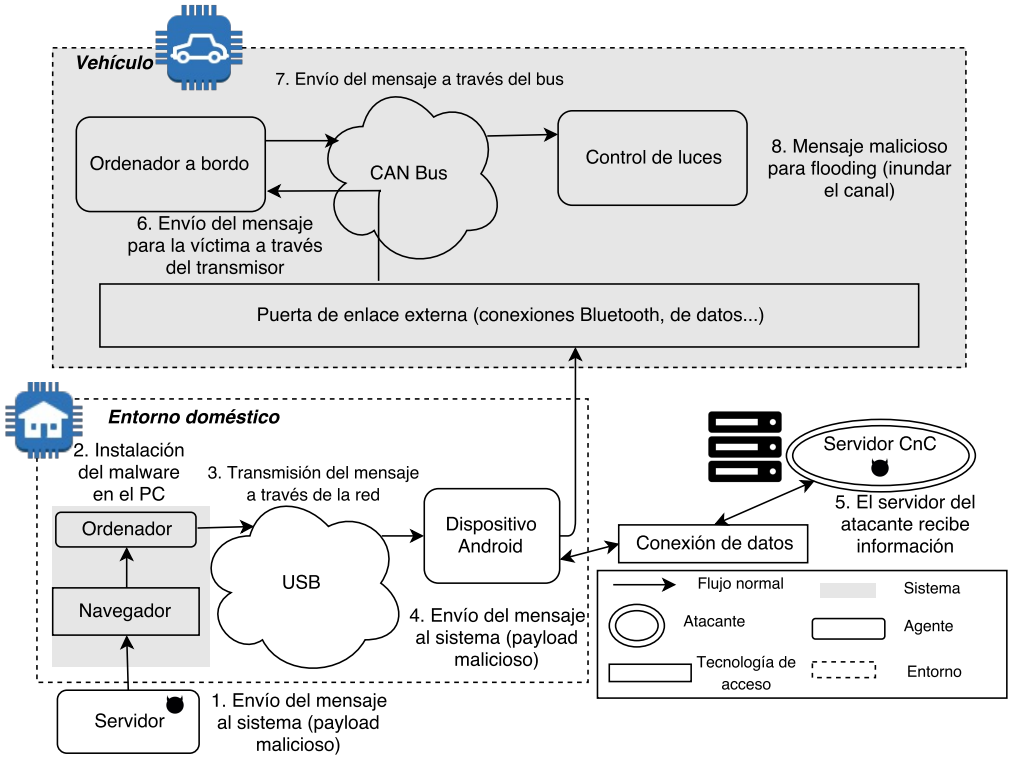

Figura 4. Aplicación del modelo BTV en un caso de uso

Tabla I
TABLA DE ABREVIATURAS PARA EL CASO DE USO

Acrónimo	**Significado**
SRV	Servidor
PC	Ordenador
AND	Dispositivo Android
OBC	Ordenador a bordo
GW	Puerta de enlace
LC	Control de luces
NVG	Navegador
DAT	Conexión de datos
NODE	Nodo
ENG	Motor

fórmula de manera innecesaria. La elección de estas abreviaturas es totalmente arbitraria y se puede extender añadiendo tantas como sea necesario, pero debe haber coherencia en usar siempre las mismas dentro de una implementación si se lleva una a cabo.

La aplicación de la fórmula que formaliza el ataque es bastante directa, dado que solamente hay que sustituir variables y funciones. Por simplicidad, se usarán abreviaturas para expresar tanto tecnologías como agentes, encontrándose éstas detalladas en la tabla I. En este caso, quedaría de la siguiente manera:

$$V = f(\alpha_1, \beta_1, \gamma_1, \rho_1, \sigma_1)_{\epsilon}^{\delta} \tag{8}$$

$$V_{CU} = a(SRV, NVG, PC, USB, AND)_{t_1}^{DAT} \tag{9}$$

Al no haber un servidor CnC en este escenario, no se incluye en la expresión previa. También se puede apreciar que, al estar analizando este escenario de manera estática, y no a lo largo del tiempo, tampoco se incluye el instante como subíndice.

V-B. Segunda fase

Una vez el smartphone haya sido infectado como se ha visto, la app instalada puede monitorizar conexiones Bluetooth con vehículos. Esto significa que al vincularse con un vehículo, puede intentar comprometer su funcionamiento. Las conexiones en el interior de los vehículos siguen el protocolo CAN (*Controller Area Network*, red de área de controladores), que rige la comunicación a través de un bus al que se conectan los distintos nodos, como pueden ser cosas tan diversas como control de ventanillas o luces, hasta dirección o motor. Debido al diseño de este protocolo, es muy sencillo realizar ataques de denegación de servicio desde cualquiera de los nodos, inundando el canal con un envío de mensajes masivo.

Hay otra clase de ataques más sofisticados, que pueden ir dirigidos a alguno de los nodos que componen la red interna del vehículo, originándose desde cualquiera de ellos. Estos ataques, sin embargo, requieren más sofisticación y un conocimiento amplio del firmware y modelo de vehículo concretos.

Tanto haciendo un ataque más sencillo como uno más complejo, se tendría un escenario en el que un smartphone Android se conecta al sistema de un automóvil mediante Bluetooth. Estas comunicaciones, en un vehículo moderno, se suelen hacer a través de una puerta de enlace para comunicaciones externas. Una vez vinculado, el smartphone podría comenzar un envío masivo de mensajes a través de esta puerta de enlace (paso 6), impidiendo así la comunicación de otros nodos con órdenes importantes, causando una denegación de servicio. Aunque algunos vehículos modernos cuentan con buses a distintos niveles para las comunicaciones críticas o con *firewalls*,

pero ha habido ataques en los que se ha logrado tumbar estos mecanismos de defensa. Otro tipo de ataque podría ser uno en el que el smartphone envía un mensaje al ordenador central para que lo redirija a otro nodo, que sería su víctima (paso 7). Este mensaje contendría un payload malicioso que afecta a la víctima, pero no al transmisor, debido a la codificación del mismo. Este nodo podría ser prácticamente cualquiera (ventanillas, cuentakilómetros, indicador de gasolina...), pero por instanciar el caso de uso, se usará el control de luces (paso 8).

Como se ha visto, el dispositivo Android pasa de ser la víctima de un ataque a ser el portador del mismo, ya que se puede transmitir a otras plataformas. En este caso, el transmisor sería la puerta de enlace de comunicaciones externas del vehículo, a la que se accede por Bluetooth, que hace las veces de tecnología de acceso. Una vez el trasmisor se vea infectado por el malware, continuará su operación con normalidad (transmitir los mensajes de los dispositivos externos que se hayan conectado), pero los demás nodos del vehículo, que tienen el papel de víctimas, y a los cuales se accede mediante el bus CAN, no podrán realizar su comunicación, ya que la inundación de paquetes por parte del smartphone habrá causado una denegación de servicio.

Como el smartphone que hace de portador, a su vez fue infectado como víctima, se puede detallar este escenario como una recursión en la fórmula. Además, al pasar los mensajes por el bus CAN desde la tecnología de acceso hasta el ordenador de a bordo, habría un cambio de función, por lo que la expresión resultante, sería la siguiente:

$$V = f(f(\alpha_1, \beta_1, \gamma_1, \rho_1, \sigma_1)^{\delta}, \beta_2, \gamma_2, \rho_2, \sigma_2)_{\epsilon} \tag{10}$$

$$V_{aCU} = b(a(SRV, NVG, PC, USB, AND)^{DAT}, GW, OBC, CAN, LC))_{t_2} \tag{11}$$

Este ataque se vale de que el estándar CAN sigue un diseño muy simple pensado para microcontroladores con poca capacidad de procesamiento, y que se centra más en comunicaciones a tiempo real. Por tanto, los mecanismos para evitar inyección de mensajes o ataques de denegación de servicio deteriorarían este tiempo de respuesta, ya que suponen operaciones más complejas.

Ya que se ha llevado a cabo la formalización del ataque observado en el caso de uso, podemos compararlo mediante la implementación detallada en la sección III-B con ataques registrados. Estos ataques son, tanto amenazas reales registradas en bases de datos de malware[13], como pruebas de concepto académicas [16].

Se puede comprobar que el ataque tiene similitudes con el troyano TROJ˙DROIDPAK.A [13], ya que usa los mismos agentes (un ordenador personal que se infecta desde una conexión externa y se transmite a un smartphone conectado por USB), y con el ataque a automóviles remoto descrito por Miller y Valasek [16], ya que la estructura es similar (mediante una conexión externa, se envían paquetes para causar que un nodo del vehículo tenga un funcionamiento distinto). Gracias a estas similitudes se puede saber qué medidas tomar para evitar su propagación o mitigar su actuación, ya que éstas están detalladas en los artículos y en las bases de datos de malware [17], tales como modificar ciertos registros de Windows o cerrar las conexiones externas a los vehículos.

Como se ha visto, el modelo BTV propuesto es extensible a varias situaciones, distintas plataformas y protocolos, y lo suficientemente versátil como para adaptarse a ataques más complejos.

VI. Discusión

Hay varios detalles a comentar en relación a los ataques Cross-platform y al enfoque seguido al desarrollar el modelo BTV. Por una parte, habría que discutir cuánto de factible es realizar un ataque de la magnitud comentada en este artículo en los sistemas actuales. Por ejemplo, para ataques sofisticados, desarrollar un firmware modificado (p. ej. para un automóvil) no es algo trivial bajo ningún concepto. Si bien en algunas pruebas de concepto, cambiar unas pocas líneas de código ha cambiado el comportamiento del vehículo de manera crucial [8], se necesitaría tener antes acceso al firmware del vehículo, lo cual no es fácil de obtener. Aunque esto se puede hacer mediante ingeniería inversa, se trata de una tarea larga y ardua, cuyo esfuerzo es difícil ver compensado. Sin embargo, la preocupación en este ámbito es creciente, como demuestran las noticias y estudios recientes al respecto [18]. Además, en relación a los ataques en plataformas de virtualización, no se debe olvidar que para tener éxito, necesitan configuraciones o versiones concretas, las cuales no se suelen tener en entornos de producción, ya que con un mínimo de mantemiento se deben llevar los parches al día.

Sin embargo, incluso teniendo en cuenta los detalles de estas situaciones y lo específicos que son los escenarios que se necesitan a veces para llevar a cabo ataques Cross-platform, no significa que estos ataques no sean factibles. Cada parte de los casos de usos mostrados se basa en casos reales de ataques vistos, ya sea en pruebas de concepto, o en la práctica, por lo que existe un riesgo real de que vayan a más y tengan consecuencias fatales para las potenciales víctimas. El modelo BTV pretende cubrir tanto estos escenarios como los nuevos que surjan. Para ello se han de definir funciones que hagan la comparación, ya sea variable a variable, o que comparen la similitud de la estructura general. En base a estas operaciones, se puede definir un porcentaje en similitud a cada uno de los ataques conocidos y saber con cuál encaja más para establecer las medidas de seguridad necesarias.

Los métodos a seguir para la búsqueda de ataques Cross-platform en bases de datos de ataques o bibliografía sobre el tema también son un punto a tener en cuenta, ya que son cruciales para la elaboración de dicha base de datos con conocimiento previo.

Por último, consideramos que este tipo de enfoque podría ser muy beneficioso para la quinta generación de redes celulares (5G), por dos motivos. Por una parte, por la gran sinergia de tecnologías software que cooperan para ofrecer servicios, empleándose multitud de plataformas diversas e integrando al usuario final (y sus dispositivos) más que nunca como parte del ecosistema. Por otra parte, por motivos operacionales y de despliegue. Este modelo y su implementación necesitan gran capacidad de recursos para su correcto funcionamiento. El procesamiento en tiempo real de los vectores de ataque requiere

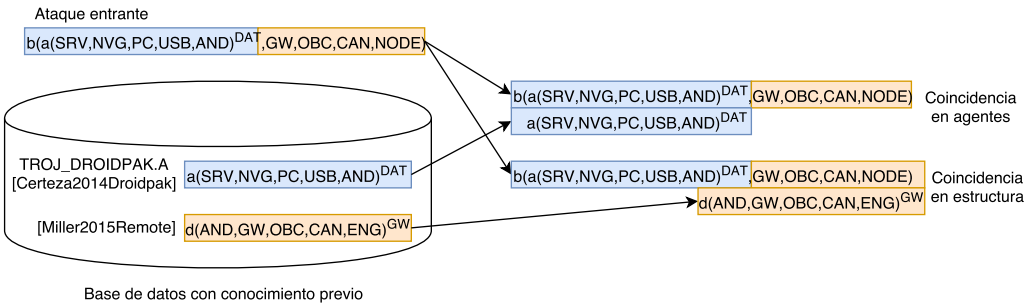

Figura 5. Comparación del ataque del caso de uso con malware conocido, utilizando la formalización propuesta

de clasificación avanzadas y manejar gran volumen de datos de forma eficiente. Estos servicios podría proporcionarlos la infraestructura 5G desde el core.

VII. Conclusiones y trabajo futuro

Los ataques Cross-platform son una realidad que se extiende cada vez más, y, conforme mejoran las infraestructuras de comunicaciones, también las posibilidades de propagación serán mayores. En este artículo se define y formaliza un modelo capaz de expresar las variaciones de los ataques cross-platform, proporcionando un enfoque especializado a cada caso, pero dentro de un mismo lenguaje para expresar los vectores de ataque y facilitar su comparación. El objetivo es ayudar en la prevención y mitigación de estas amenazas, permitiendo que los componentes intermedios puedan predecir si algo que está ocurriendo podría afectar a otros sistemas con los que se interrelacionan.

En lo relativo al trabajo futuro y mejoras, la aplicación directa y práctica de este trabajo pasa por modelar, empleando BTV, un conjunto de arquitecturas y sistemas que hayan sido objeto de ataques Cross-platform, y analizar sus relaciones para encontrar posibles similitudes y observar si se puede obtener información sobre vulnerabilidades de los mismos en base a ellas.

Además, se analizarán los ataques Cross-platform conocidos, y se procederá a su formalización con el modelo propuesto, para tener una base de conocimiento lo suficientemente representativa con la que elaborar un sistema de detección y prevención que ayude a mitigar el efecto de ataques de esta naturaleza. Un entorno en el que puede ser particularmente útil este sistema es 5G, debido a la naturaleza software y multi-capa en la que se basan sus comunicaciones.

VIII. Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Economía y Competitividad a través de los proyectos IoTest (TIN2015-72634-EXP) y SMOG (TIN2016-79095-C2-1-R). El segundo autor ha sido financiado por INCIBE a través del programa de ayudas para la excelencia de los equipos de investigación avanzada en ciberseguridad.

Referencias

[1] M. Lindorfer, M. Neumayr, J. Caballero, and C. Platzer, "Poster: Cross-platform malware: write once, infect everywhere," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 1425–1428.

[2] Eugene, "Architecture spanning shellcode," 2000.

[3] K. Chen, X. Wang, Y. Chen, P. Wang, Y. Lee, X. Wang, B. Ma, A. Wang, Y. Zhang, and W. Zou, "Following devil's footprints: Cross-platform analysis of potentially harmful libraries on android and ios," in *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 2016, pp. 357–376.

[4] C. R. Reeves Jr, "Cross platform network authentication and authorization model," Feb. 13 2007, uS Patent 7,178,163.

[5] W. Jia, D. Bin, and L. Liao, "Architecture of secure cross-platform and network communications," in *Proceedings of the 2nd international conference on Ubiquitous information management and communication*. ACM, 2008, pp. 321–328.

[6] K. Kotapati, P. Liu, Y. Sun, and T. LaPorta, "A taxonomy of cyber attacks on 3g networks," *Intelligence and Security Informatics*, pp. 129–138, 2005.

[7] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno *et al.*, "Comprehensive experimental analyses of automotive attack surfaces." in *USENIX Security Symposium*. San Francisco, 2011.

[8] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 447–462.

[9] N. Elhage, "Virtunoid: Breaking out of kvm," *Black Hat USA*, 2011.

[10] K. Kortchinsky, "Cloudburst: A vmware guest to host escape story," *Black Hat USA*, 2009.

[11] M. Mimoso, "Windows botnet spreading mirai variant," 2017.

[12] "Android.ciaco," 2013.

[13] R. Certeza, "Cross-platform mobile threats: A multi-pronged attack," 2014.

[14] X. Gui, J. Liu, M. Chi, C. Li, and Z. Lei, "Analysis of malware application based on massive network traffic," *China Communications*, vol. 13, no. 8, pp. 209–221, 2016.

[15] M. Sargent, "Malware trends: The rise of cross-platform malware," 2012.

[16] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, 2015.

[17] G. R. Joi, "Troj˙droidpak.a," 2014.

[18] F. Maggi, "The crisis of connected cars: When vulnerabilites affect the can standard," 2017.