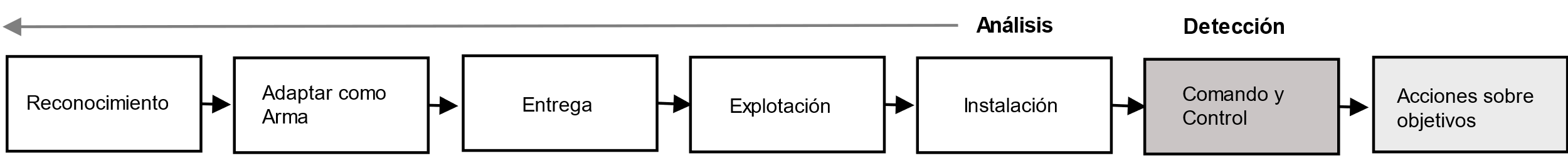
Un ciberataque se define como un “intento de destruir, exponer, alterar, inhabilitar, robar u obtener acceso no autorizado o hacer un uso no autorizado de un activo”[[1]](#footnote-1). El estudio de los ciberataques está adquiriendo relevancia a nivel social debido al continuo avance en la automatización, que es transversal a todas las organizaciones, haciendo a los seres humanos cada vez más dependientes de la tecnología. Al mismo tiempo, los ataques han proliferado en los últimos años en Chile. Por ejemplo, se pueden destacar los casos del Banco de Chile, Banco Estado, Cencosud, ClaveÚnica y, más recientemente, Sonda.

En algunos de estos casos, se debió detener gran parte de su operación, mientras que en otros solo se sufrió daño reputacional y de confianza por parte de los ciudadanos. Como consecuencia, las organizaciones privadas y públicas de cierto tamaño disponen de unidades dedicadas a mitigar los daños producidos por estos ataques. Estas unidades son conocidas como Centros de Operación de Seguridad (SOC). Entre las herramientas con las que cuentan estos SOC se encuentran los sistemas de detección de intrusos (IDS), las cuales generan alertas frente a eventos sospechosos en las redes informáticas que auditan. A inicios del 2000, solo el 1 % de las alertas generadas por un IDS correspondían a ataques reales[[2]](#footnote-2)*.*

Un tipo de ciberataque difícil de detectar son los ataques de red de múltiples etapas o múltiples pasos (MSNA). Estos consisten en un conjunto de ataques que se realizan para alcanzar una meta específica que contiene al menos dos ataques distintos relacionados[[3]](#footnote-3)*.* Hasta alcanzar su meta, estos ataques pueden durar horas, días o meses. Los IDS solo detectan cada paso como un ataque aislado, posiblemente de bajo riesgo, y lo alertan de esta manera. Normalmente, la víctima de una MSNA es una organización con un alto grado de seguridad sobre una vigilada y compleja topología de red. Por ello, los atacantes los utilizan para mantener el sigilo. El primer marco de trabajo para este tipo de ataque fue el Cyber Kill Chain (CKC) de Hutchins et al.[[4]](#footnote-4), el cual se basa en la táctica Kill Chain del F2T2EA usado por el ejército de los Estados Unidos (EE. UU.) (encontrar, arreglar, rastrear, apuntar, atacar y evaluar).

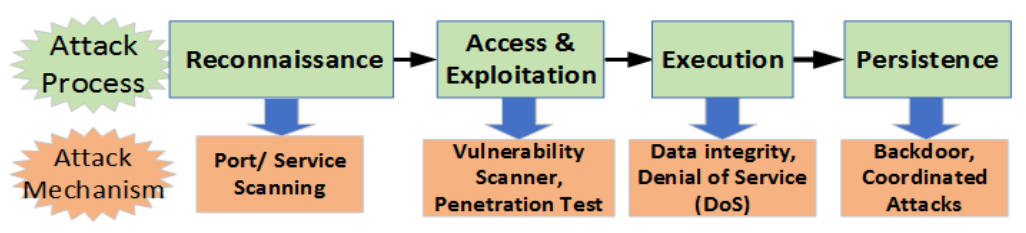
**Figura 1**. Cyber Kill Chain (CKC)



Fuente: HUTCHINS, Eric, CLOPPERT, Michael y AMIN, Rohan. 2011. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Leading Issues in Information Warfare & Security Research [en línea], vol. 1, no. 1, 1-14. Disponible en: https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf [Consulta: 29 de agosto de 2021].

Este CKC está orientado a desactivar amenazas del tipo “Amenaza Avanzada Persistente” (APT), para lo cual recopila inteligencia sobre las amenazas y su uso futuro, permitiendo que el atacante continúe en sus actividades después de que lo detecten[[5]](#footnote-5).

**Figura 2**. CKC simplificado con procesos y mecanismos de ataque



Fuente: SINGH, Vivek, CALLUPE, Steven y GOVINDARASU, Manimaran. Testbed-based Evaluation of SIEM Tool for Cyber Kill Chain Model in Power Grid SCADA System. En: 2019 North American Power Symposium (NAPS) [en línea]. Kansas, Estados Unidos, octubre de 2019, pp. 1-6. Disponible en: https://www.researchgate.net/publication/336623025\_Testbed-based\_Evaluation\_of\_SIEM\_Tool\_for\_Cyber\_Kill\_Chain\_Model\_in\_Power\_Grid\_SCADA\_System [Consulta: 29 de agosto de 2021].

En la misma línea, Singh et al.[[6]](#footnote-6) propusieron un modelo simplificado del CKC compuesto de cuatro pasos: reconocimiento, acceso, ejecución y persistencia. La definición de cada uno de estos pasos es la siguiente:

1. Reconocimiento: el atacante trata de recopilar información relevante sobre la red y sus servicios para identificar posibles víctimas y objetivos de ataque. Para ello, puede usar herramientas de red como ping, arp, traceroute, nmap, entre otras.
2. Acceso: el atacante intenta conectarse a los objetivos seleccionados para descubrir sus posibles vulnerabilidades. Esta información se usará más adelante para lograr acceso y/o escalada de privilegios. En esta etapa, pueden utilizarse herramientas como OpenVAS, Metasploit, Nessus, entre otras.
3. Lanzamiento/ejecución del ataque: en esta etapa, el atacante intenta explotar las vulnerabilidades encontradas en la red o sus componentes para obtener control sobre ellos. Las actividades pueden incluir virus, gusanos, caballos de Troya, ataques de denegación de servicio (DoS), ataques de Man-in-the-Middle (MITM), violaciones a la integridad y violaciones a la privacidad.
4. Persistencia: en esta etapa, el atacante crea una puerta trasera para mantener su acceso persistente y poder ingresar en el futuro para repetir el ataque o lanzar múltiples ataques en diferentes plataformas de manera coordinada. La persistencia requiere que el lanzamiento/ejecución se haya realizado con éxito.

En esta investigación se utilizó el CKC simplificado, debido a que está compuesto por pasos fácilmente reconocibles desde una bitácora o registro de tramas de red de un ataque MSNA.

Existen diversas propuestas de autores para detectar los MSNA[[7]](#footnote-7), aunque pocas de ellas utilizan aprendizaje automático o *deep learning*. Eso se debe principalmente a la escasez de conjuntos de datos de MSNA para entrenar algoritmos, dado que estos suelen estar obsoletos o tener una cantidad reducida de ejemplos de MSNA. Por ejemplo, se han presentado conjuntos de datos que incluyen siete escenarios de ataque diferentes[[8]](#footnote-8) (fuerza bruta, Heartbleed, Botnet, DoS, DDoS, ataques web e infiltración de la red desde adentro) en una infraestructura que consta de 50 máquinas; otro conjunto de datos que contiene el comportamiento abstracto de 25 usuarios basados en los protocolos HTTP, HTTPS, FTP, SSH y de correo electrónico de cinco días, con ataques diversos (fuerza bruta FTP, fuerza bruta SSH, DoS, Heartbleed, ataque web, infiltración, Botnet y DDoS)[[9]](#footnote-9); y uno muy utilizado que incluye tres escenarios específicos[[10]](#footnote-10).

Por otro lado, los ejercicios de ciberseguridad tipo Capture the Flag (CTF) han tomado gran relevancia en la difusión de esta temática. En este tipo de eventos, los equipos de participantes utilizan sus conocimientos de manera práctica en ciberseguridad resolviendo diferentes tareas en un ambiente de aprendizaje en línea, tales como: explotar vulnerabilidades de sitios web, descubrir claves, ingresar a redes con brechas de seguridad, entre otros. Típicamente, la bandera rescata un texto que está escondido en la máquina víctima. Hay tres tipos comunes de CTF: Jeopardy, Ataque-Defensa y mixtos.

Los CTF de estilo Jeopardy tienen tareas en una variedad de categorías, tales como Web, Forense, Crypto, Binary, entre otras. El equipo puede ganar puntos por cada tarea resuelta.

En los CTF de Ataque-Defensa, también llamados juegos de guerra, cada equipo tiene su propia red y/o *host*, con servicios vulnerables. Cada equipo tiene un tiempo corto para parchar sus servicios y desarrollar *exploits* para luego conectarse con los otros equipos participantes. En ese momento, cada equipo debe proteger sus propios servicios para obtener puntos de defensa y lograr romper la seguridad de los oponentes para recibir puntos de ataque. El ganador de la competencia es el equipo con la mayor cantidad de puntos.

El DEF CON CTF es el CTF de mayor duración de todos los tiempos. Este es del tipo Ataque-Defensa y las finales son de gran dificultad. Este CTF ha existido desde DEF CON 4 (1996). Su formato actual comenzó en DEF CON 10 (2002). A partir de entonces, se refinó aún más en DEF CON 13 (2005) cuando se enfatizó en la explotación binaria y su corrección. Finalmente, desde DEF CON 25 (2017), se utiliza el emulador cLEMENCy (LEgitbs Middle ENdian Computer) desde el cual se ejecutan los servicios vulnerables para que los participantes estén obligados a prescindir de herramientas avanzadas de corrección de *software* que podrían favorecer a algún equipo en particular. En la actualidad, estos CTF se utilizan para entrenar profesionales en seguridad de una manera lúdica, tanto en ambientes educacionales[[11]](#footnote-11) como en gobiernos y en compañías privadas. Un ejemplo de esto último son los organizados por compañías como Google en 2020 (https://capturetheflag.withgoogle.com/) y Facebook en 2019 (https://www.facebook.com/notes/facebook-bug-bounty/announcing-facebook-ctf-2019/2629218463759030/).

La motivación de este trabajo es construir un método para caracterizar ataques multietapas en CTF. Esto se debe a que los MSNA ocurren de manera natural en estos ambientes controlados. Además, los equipos están altamente preparados y utilizan técnicas sofisticadas, lo que los hace muy interesantes y potencialmente reproducibles y auditables debido a que suelen registrarse.

Respecto de la reconstrucción de ataques, en la literatura se han utilizado meta-alertas correlacionadas, así como alertas únicas no agrupadas para construir grafos de escenarios de APT[[12]](#footnote-12). Esto se logra construyendo una Máquina de Estados de Kill Chain (KCSM) que opera con datos de alerta agrupados para identificar estados y transiciones de ataques de múltiples etapas. Los grafos de escenarios de APT resultantes de este proceso visualizan posibles campañas de APT en la red y proporcionan un contexto procesable durante las investigaciones. Otro enfoque ha sido utilizar reconocimiento de escenarios de ataque basados en pasos de ataque, lo que permite ofrecer un enfoque flexible para reconocer ataques[[13]](#footnote-13)*.* El sistema que proponen puede aportar un valor añadido en investigaciones forenses y en *honeypots* de investigación. Su arquitectura usa seis componentes principales: receptor de alertas, normalización de alertas, preprocesamiento de alertas, agrupación de alertas, reducción de alertas y reconocimiento de escenarios de ataque.

En este trabajo se propone un método para la caracterización de ataque multietapa en ejercicios CTF. Este método utiliza como base un IDS, en este caso Snort, para generar un conjunto de alertas. Las alertas son procesadas, clasificadas en las diferentes etapas del modelo CKC y correlacionadas para caracterizar visualmente los MSNA. Este trabajo se divide en las siguientes secciones: Sección II: “Método propuesto para la caracterización de un CTF”; Sección III: “Resultados”, donde se muestran los resultados aplicando este método al CTF 22 de la DEF CON; y Sección IV, donde se establecen las conclusiones y el potencial trabajo futuro.

**II. MÉTODO PROPUESTO PARA LA CARACTERIZACIÓN DE UN CT**

En esta investigación se propuso utilizar el método Kill Chain Simplificado en Máquinas de Estado (KCS2ME), es decir, emplear máquinas de estado para representar los pasos del modelo de CKC simplificado visto anteriormente.

**Figura 3**. Método general de caracterización de ataques multietapa en ejercicios CTF

Diagrama

Descripción generada automáticamente

Fuente: elaboración propia.

La Figura 3 muestra la propuesta para caracterizar ataques multietapa en ejercicios CTF utilizando un enfoque basado en alertas en vez de las tramas directas de la red. Esta propuesta consta de dos bloques principales, el primero es la operación de Snort de manera normal sobre las capturas de red y el segundo es la clasificación de las alertas obtenidas según el modelo CKC. La primera parte, que implica la operación normal de Snort, requiere simplemente la selección del CTF a analizar y genera alertas a partir de las capturas de las tramas de red registradas durante el CTF. La segunda requiere el apoyo del *software* (https://github.com/communitylab4u/MSNA/) y la clasificación previa de las reglas de Snort a utilizar mediante el modelo CKC, que se encuentran disponibles con el *software*. Luego, se ejecuta la clasificación de las alertas y la caracterización visual de los CTF.

Por lo tanto, en el bloque de la izquierda se muestra el primer componente del método: Snort, un Sistema Detector de Intrusos de Red (NIDS), el cual es de fuente abierta, no requiere un pago por licencia y viene integrado con Linux Ubuntu. La entrada al componente Snort incluye las reglas Snort actualizadas previamente en su sitio web y los archivos que contienen las capturas de las tramas de red de CTF, que generalmente están disponibles en formato PCAP (por ejemplo: https://defcon.org/html/links/dc-ctf.html). Para procesar el *dataset*, se ejecuta el siguiente comando:

snort -c /etc/snort/snort.conf -r archivo.pcap

Por otro lado, en el bloque derecho de la Figura 3 se pueden observar dos componentes adicionales: Base de Conocimiento y *Software* MSNA (repositorio de clasificación de alertas y conjunto de scripts que se encuentran disponibles en https://github.com/communitylab4u/MSNA/). Para el componente Base de Conocimiento, dado que las reglas de Snort son generales, una parte del método consiste en clasificar las reglas en las etapas del modelo CKC, tal como se muestra en la Figura 4 y se describe en el Algoritmo 1. Para ello, primero se genera una plantilla vacía utilizando un *script* que toma las reglas instaladas dentro de Snort, a partir del cual se extrae el SID de cada regla y se generan los campos necesarios para el siguiente paso.

**Algoritmo 1**. Algoritmo utilizado para la clasificación CKC de las alertas de los *dataset*

Texto

Descripción generada automáticamente

Fuente: elaboración propia.

**Figura 4**. Generación de la Base de Conocimiento usando los métodos de clasificación

Diagrama

Descripción generada automáticamente

Fuente: elaboración propia.

El componente Base de Conocimientose refiere al repositorio que almacena tuplas de regla-etapa. Las técnicas utilizadas para realizar la clasificación de las reglas Snort, dentro de una etapa del CKC, son las siguientes:

* **Basado en el código de la vulnerabilidad**: muchas reglas incluyen en su descripción un código de bases de datos de vulnerabilidades como CVE, bugtraq, msb, arachNIDS, osvdb, McAfee, Nessus o una URL que permite obtener información acerca de lo que la regla busca detectar y así clasificarla en el CKC. Un ejemplo de esto se puede ver en la regla 100000223, cuya descripción es *reference:bugtraq,16213*, lo que permite encontrar información en https://www.securityfocus.com/bid/16213/.
* **Basado en la definición de la regla:** la regla Snort contiene las condiciones en las que debe alertar algún tipo de ataque, y al interpretar estas condiciones se puede hacer la correspondiente clasificación en el modelo CKC, por ejemplo:

alert tcp $EXTERNAL\_NET any - $HTTP\_SERVERS $HTTP\_PORTS (msg:“WEBATTACKS nmap command attempt”, flow: to\_server, established, content: “nmap%20”, nocase, classtype: web application attack, sid:1361, rev:5,)

Allí se indica que se debe generar una alerta cuando un cliente se identifica a sí mismo como “nmap” al conectarse desde el exterior hacia el servidor web, lo que puede interpretarse como un intento de escanear servicios web para obtener información sobre la versión del servidor. Esto se considera como una etapa 2 del modelo CKC.

* **Basado en la descripción**: en la descripción de la regla puede haber un texto que indica de manera certera su naturaleza. Los más importantes son: XSS, Overflow, SQL injection, exploit.
* **Basado en el uso de herramientas**: esto corresponde al uso de herramientas que se sabe qué tipo de resultados generan. Por ejemplo, haciendo reconocimiento de red, usando ping o traceroute, generará alertas que se pueden clasificar como de descubrimiento de red, es decir, de etapa 1 del CKC usado.
* **Basado en el nombre del archivo**: las reglas de Snort se agrupan en archivos por temática, lo que es útil en algunos casos, mientras que en otros no. Entre los nombres de archivo que pueden ayudar a clasificar se encuentran los siguientes: Virus, Backdoor y Shellcode. Estos implican compromiso del *host*, es decir, etapa 4 del CKC. En cambio, otros nombres como Web o Misc no permiten hacer una adecuada clasificación.

Una vez que el *software* MSNA procesa las tramas de red de un CTF, se generará un conjunto de alertas clasificadas en el modelo CKC. La salida incluirá un archivo con los Snort ID (SID) de las reglas que no estaban clasificadas en el CKC, además de la cantidad de veces que se encontraron en el archivo de alertas.

Para ello, se realiza el supuesto de que el destino del ataque es un servidor o un grupo de servidores en producción y, por tanto, no tienen usuarios haciendo uso de él en su consola, como en el caso de un equipo de escritorio. Esto implica lo siguiente: 1. Las conexiones salientes son automatizadas, por lo que normalmente una conexión saliente será exitosa y las credenciales de usuario y contraseña serán correctas. Otros comportamientos, como un barrido al exterior, se consideran bajo el control de un atacante; 2. No hay sistemas de chat en uso. Por esta razón, una conexión de chat de salida se considerará una acción de comando y control; 3. Una conexión saliente de un virus implica que el servidor está comprometido; y 4. Una conexión desde el exterior a un servicio del servidor que debería entregarse solo a equipos de la misma LAN se considerará un ataque. Ejemplos de estos servicios son compartir carpetas e impresoras usando SMB, así como el servicio de proxy. En este último caso, al estar el cliente fuera de la LAN, se considera que esta conexión es una búsqueda de proxy reverso para aprovecharlos de alguna manera (por ejemplo, un ataque de falsa bandera o un salto hacia el interior de la organización).

1. INTERNATIONAL ORGANIZATION FOR STANDARDIZATION [ISO]. ISO/IEC. 27000:2016. Information technology — Security techniques — Information security management systems — Overview and vocabulary [en línea]. 2016. Disponible en: https://www.iso.org/standard/66435.html [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-1)
2. JULISCH, Klaus. 2003. Clustering intrusión detection alarms to support root cause analysis. ACM Transactions on Information and System Security [en línea], vol. 6, no. 4, 443-471. Disponible en: https://doi.org/10.1145/950191.950192 [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-2)
3. NAVARRO, Julio, DERUYVER, Aline y PARREND, Pierre. 2018. A systematic survey on multi-step attack detection. Computers Security [en línea], vol. 76, 214-249. Disponible en: https://doi.org/10.1016/j.cose.2018.03.001 [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-3)
4. HUTCHINS, Eric, CLOPPERT, Michael y AMIN, Rohan. 2011. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Leading Issues in Information Warfare & Security Research [en línea], vol. 1, no. 1, 1-14. Disponible en: https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-4)
5. BUKAC, Vit, LORENC, Vaclav y MATYÁŠ, Vashek. 2014. Red Queen 's Race: APT Win-Win Game [en línea]. En: CHRISTIANSON, B., MALCOLM, J., MATYÁŠ, V., ŠVENDA, P., STAJANO, F. y ANDERSON, J. *Security Protocols XXII.* *Security Protocols 2014. Lecture Notes in Computer Science*. Suiza: Springer, Cham, 55-61. Disponible en: https://doi.org/10.1007/978-3-319-12400-1\_7 [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-5)
6. SINGH, Vivek, CALLUPE, Steven y GOVINDARASU, Manimaran. Testbed-based Evaluation of SIEM Tool for Cyber Kill Chain Model in Power Grid SCADA System. En: 2019 North American Power Symposium (NAPS) [en línea]. Kansas, Estados Unidos, octubre de 2019, pp. 1-6. Disponible en: https://www.researchgate.net/publication/336623025\_Testbed-based\_Evaluation\_of\_SIEM\_Tool\_for\_Cyber\_Kill\_Chain\_Model\_in\_Power\_Grid\_SCADA\_System [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-6)
7. NAVARRO, Julio, DERUYVER, Aline y PARREND, Pierre. 2018. A systematic survey on multi-step attack detection, op. cit. [↑](#footnote-ref-7)
8. UNIVERSITY OF NEW BRUNSWICK [UNB]. CSE-CIC-IDS2018 on AWS. A collaborative project between the Communications Security Establishment (CSE) & the Canadian Institute for Cybersecurity (CIC) [en línea]. 2020a. Disponible en: https://www.unb.ca/cic/datasets/ids-2018.html [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-8)
9. UNIVERSITY OF NEW BRUNSWICK [UNB]. Intrusion detection evaluation dataset (CIC-IDS2017) [en línea]. 2020b. Disponible en: https://www.unb.ca/cic/datasets/ids-2017.html [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-9)
10. HAINES, Joshua. 2000 Darpa intrusión detection scenario specific data sets [en línea]. 2020. Disponible en: https://archive.ll.mit.edu/ideval/data/2000data.html [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-10)
11. MIRKOVIC, J. y PETERSON, P. 2014. Class Capture-the-Flag Exercises. En: 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14) [en línea]. San Diego, California, agosto de 2014. Disponible en: https://www.usenix.org/biblio/class-capture-flag-exercises [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-11)
12. WILKENS, Florian et al. Multi-Stage Attack Detection via Kill Chain State Machines [en línea]. 2021. Disponible en: https://arxiv.org/abs/2103.14628 [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-12)
13. MENDES, J. y SOARES, R. 2019. Flexible Approach to Multi-Stage Network Attack Recognition. International Journal of Computer Science and Information Security (IJCSIS) [en línea], vol. 17, no. 8, 67-73. Disponible en: https://www.academia.edu/40458556/Flexible\_Approach\_to\_Multi\_Stage\_Network\_Attack\_Recognition [Consulta: 29 de agosto de 2021]. [↑](#footnote-ref-13)