Jornadas de Automática

Diseño de un sistema de vulnerabilidades dockerizado en 5G

Diego Narciandi-Rodríguez[a,*], Jose Aveleira-Mata[b], María Teresa García-Ordás[d], Hugo Bernardo-Garmilla[a], Javier Alfonso-Cendón[c], Isaías García-Rodríguez[d]

[a]*RIASC. Research Institute of Applied Sciences in Cybersecurity. Universidad de León, MIC. Campus de Vegazana, León, 24071, Spain*
[b]*SECOMUCI Research Group. Department of Electric, Systems and Automatics Engineering. Universidad de León, Escuela de Ingenierías. Campus de Vegazana, León, 24071, Spain*
[c]*Department of Mechanical, Computer Science and Aerospace Engineering. Universidad de León, Escuela de Ingenierías. Campus de Vegazana, León, 24071, Spain*
[d]*ALBA Research Group. Department of Electric, Systems and Automatics Engineering. Universidad de León, Escuela de Ingenierías. Campus de Vegazana, León, 24071, Spain*

To cite this article: Narciandi-Rodríguez, D, Aveleira-Mata, J, García-Ordás, MT, Bernardo-Garmilla, H, Alfonso-Cendón, J, García-Rodríguez, I. 2026. Design of a Dockerised Vulnerability System over 5G. Jornadas de Automática, 47. https://doi.org/10.17979/ja-cea.2026.47.13829

Resumen

Este trabajo presenta el diseño de un banco de pruebas para la generación controlada de tráfico de ciberseguridad en redes privadas 5G. La propuesta combina una infraestructura Firecell Labkit 40, conectividad radio y un equipo que aloja servicios vulnerables contenerizados basados en Docker-Vulhub. El entorno despliega vulnerabilidades conocidas, muchas documentadas mediante CVE, hace circular el tráfico por la red 5G SA y lo captura desde el core mediante herramientas como `tcpdump`. Una prueba de concepto que captura desde el core un ataque real contra Apache Tomcat (CVE-2017-12615) demuestra que el tráfico es observable y reconstruible, y que de él puede derivarse un dataset etiquetado. La arquitectura es una base modular y ampliable para futuras campañas de creación de datasets. Su objetivo no es evaluar exhaustivamente cada vulnerabilidad, sino documentar un entorno reproducible para generar tráfico útil en la evaluación de IDS y modelos de aprendizaje automático en 5G privado.

Palabras clave: Sistemas de control en red seguros, Seguridad, Monitorización, Diseño experimental

Design of a Dockerised Vulnerability System over 5G

Abstract

This paper presents the design of a testbed for the controlled generation of cybersecurity traffic in private 5G networks. The proposal combines a Firecell Labkit 40 infrastructure, radio connectivity, and a host machine running Dockerised vulnerable services based on Docker-Vulhub. The environment deploys known vulnerabilities, many of them documented through CVEs, routes the resulting traffic through the 5G SA network, and captures it from the core using tools such as `tcpdump`. A proof of concept that captures a real attack against Apache Tomcat (CVE-2017-12615) from the core shows that the traffic is observable and reconstructable, and that a labelled dataset can be derived from it. The architecture is a modular and extensible basis for future dataset-generation campaigns. Its main objective is not to exhaustively evaluate each vulnerability, but to document a reproducible environment for generating useful traffic to evaluate IDS and machine learning models in private 5G networks.

Keywords: Secure networked control systems, Security, Monitoring, Experiment design

*Autor para correspondencia: dnarr@unileon.es

1. Introducción

La quinta generación de redes móviles (5G) se ha consolidado como tecnología habilitadora de escenarios de conectividad avanzada, baja latencia y alta densidad de dispositivos. Más allá de los operadores públicos, cada vez más empresas, centros de investigación y administraciones despliegan redes 5G privadas para comunicaciones críticas, servicios IoT y sistemas de control y monitorización remota, donde una intrusión puede degradar la disponibilidad o integridad del servicio. Esta adopción, sin embargo, no elimina los riesgos: la coexistencia de dispositivos IoT, funciones de red virtualizadas, servicios expuestos y arquitecturas basadas en software amplía la superficie de ataque (configuraciones inseguras, denegación de servicio, explotación de aplicaciones o anomalías en los planos de control y de datos) (Narciandi-Rodríguez et al., 2025a).

Frente a este contexto, los sistemas de detección de intrusiones (IDS) y el aprendizaje automático son una línea de defensa prometedora, pero su entrenamiento y validación exigen datos representativos, bien capturados y, a ser posible, etiquetados. En 5G esta necesidad es crítica: el comportamiento del tráfico no solo depende de la aplicación atacada, sino de cómo este atraviesa la arquitectura móvil, con elementos del core, interfaces internas y protocolos como GTP, SCTP o NGAP.

Sin embargo, la disponibilidad de datasets de ciberseguridad generados sobre redes 5G privadas sigue siendo limitada. Los conjuntos ampliamente utilizados para evaluar IDS (KDD, UNSW-NB15, CICIDS2017 o CSE-CIC-IDS2018) no se generaron sobre infraestructuras 5G y no capturan características propias de estas redes, como la señalización NGAP, las asociaciones SCTP, el tráfico de usuario tunelizado por GTP-U o los procedimientos de sesión PDU (Tavallaee et al., 2009; Moustafa and Slay, 2015; Sharafaldin et al., 2018; Thakkar and Lohiya, 2020). Datasets más recientes orientados a IoT, IIoT y edge, como Bot-IoT, TON_IoT o Edge-IIoTset, aportan escenarios más actuales, pero tampoco reproducen el comportamiento de una red 5G SA ni la observabilidad desde el core (Koroniotis et al., 2019; Alsaedi et al., 2020; Ferrag et al., 2022).

Se han propuesto recursos más próximos al 5G, como 5GAD-2022, 5G-NIDD o trabajos sobre detección de DoS/DDoS en slices y funciones del core (Coldwell et al., 2022; Siriwardhana et al., 2025; Khan et al., 2022; Amponis et al., 2022). No obstante, muchos se centran en ataques concretos, usan entornos simulados o no detallan la relación entre topología, servicios, dispositivos y punto exacto de captura, lo que dificulta comparar trabajos, reproducir resultados y evaluar IDS en condiciones próximas a un despliegue real (Goldschmidt and Chudá, 2025; Hamroun et al., 2025; Noor et al., 2025).

La creación de datasets en redes 5G privadas añade una barrera experimental: requiere equipamiento específico (core, acceso radio, SIM y dispositivos) y, para el tráfico malicioso, servicios vulnerables y mecanismos de etiquetado. Trabajos previos han capturado tráfico en entornos 5G SA reales para estudiar condiciones anómalas, como ataques de jamming (Narciandi-Rodríguez et al., 2026; Narciandi-Rodríguez et al., 2025b).

Este trabajo propone una arquitectura experimental para facilitar la creación de datasets de ciberseguridad en redes privadas 5G, combinando servicios vulnerables en contenedores basados en Docker-Vulhub, una infraestructura privada Firecell Labkit 40 y un punto de captura en el core 5G. Su principal contribución es la definición y documentación de un entorno modular, ampliable y reproducible: frente a las aproximaciones basadas solo en simulación o en capturas aisladas, incorpora, retira o sustituye servicios de forma controlada, lo que facilita construir escenarios de ataque progresivos y generar datasets etiquetables para evaluar IDS, modelos de aprendizaje automático y mecanismos de supervisión en redes 5G privadas que soportan automatización y control remoto. Como primera validación, se incluye una prueba de concepto que captura desde el core un ataque real y deriva de él un conjunto de datos etiquetado.

El resto del trabajo se organiza como sigue. La Sección 2 presenta los métodos y materiales, incluyendo el equipamiento empleado, la arquitectura del banco de pruebas, su despliegue, la validación operativa y el procedimiento de generación y captura de tráfico. La Sección 3 detalla la selección de servicios vulnerables. La Sección 4 presenta una prueba de concepto de captura desde el core 5G. La Sección 5 discute el alcance, las aportaciones y las limitaciones, y la Sección 6 recoge las conclusiones y las líneas de trabajo futuro.

2. Métodos y materiales

2.1. Materiales empleados

El banco de pruebas se ha construido a partir de una combinación de equipamiento 5G privado, hardware de conectividad, radio definida por software y herramientas de contenerización. Los principales elementos empleados son los siguientes:

- **Firecell Labkit 40**: plataforma privada 5G basada en OpenAirInterface (OAI), operada en modo 5G SA (*Standalone*). Se emplea como infraestructura 5G de laboratorio: conecta equipos mediante SIM y captura tráfico desde el core con herramientas como `tcpdump`.
- **USRP B210**: transceptor de radio definida por software que proporciona el enlace radio 5G del despliegue Firecell.
- **Cradlepoint R1900**: router 5G que actúa como pasarela entre el MiniPC y la red privada 5G; incorpora una SIM del core Firecell y se conecta al MiniPC por Ethernet.
- **MiniPC Blackview MP100 Pro**: equipo anfitrión que aloja los servicios vulnerables, con Windows 11 y Docker Desktop.
- **Docker Desktop y Docker Compose**: despliegan, detienen y reconstruyen los servicios de forma reproducible; el archivo `docker-compose.yml` define los contenedores, los puertos expuestos y la red Docker común.
- **Docker-Vulhub**: repositorio de entornos vulnerables, base de los servicios desplegados, con escenarios asociados a vulnerabilidades conocidas, muchas documentadas mediante CVE.

- **Red Docker** `vuln-net`: red común que integra los contenedores vulnerables bajo una misma estructura lógica.
- **Script PowerShell de validación**: comprueba el estado de Docker, enumera los contenedores activos y verifica la accesibilidad de los servicios por HTTP o TCP.

Esta combinación separa el plano de aplicación, donde se ejecutan los servicios vulnerables, del plano de red, donde se observa el tráfico: las comunicaciones hacia los contenedores atraviesan la red privada 5G antes de capturarse en el core.

2.2. *Arquitectura general del banco de pruebas*

La arquitectura se ha diseñado para generar de forma controlada tráfico de ciberseguridad en una red privada 5G: despliega servicios vulnerables en contenedores, los hace accesibles a través de la infraestructura 5G y captura el tráfico resultante desde el core, con vistas a crear datasets etiquetables para evaluar IDS y modelos de aprendizaje automático.

La arquitectura se organiza en tres bloques: (i) el entorno de servicios vulnerables, contenedores Docker basados en Vulhub (Vulhub, 2026); (ii) el MiniPC que los aloja; y (iii) la infraestructura de comunicaciones 5G: el router Cradlepoint R1900 y la red privada Firecell Labkit 40 en modo 5G SA, donde se captura el tráfico en el core.

Cada contenedor se publica en el anfitrión mediante un mapeo de puertos (*port mapping*) a un puerto externo distinto, lo que permite seleccionar el servicio objetivo de cada experimento sin modificar la red. Las comunicaciones dirigidas a los servicios atraviesan la infraestructura móvil a través del Cradlepoint, y el core 5G actúa como punto de observación, donde se captura el tráfico con `tcpdump`.

La Figura 1 muestra una visión general del banco de pruebas. En ella se representa la relación entre el entorno de contenedores, el nodo físico de despliegue, el router 5G y el core de la red privada. Aunque la arquitectura permite incorporar posteriormente terminales externos de prueba conectados a la misma red 5G, este trabajo se centra en la definición y documentación del entorno experimental.

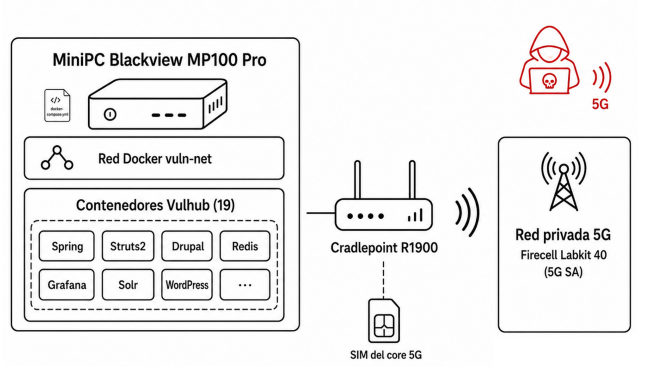

Figura 1: Arquitectura general del banco de pruebas propuesto para la generación de tráfico de ciberseguridad en una red privada 5G.

2.3. *Despliegue del entorno*

El despliegue se realiza desde una carpeta principal del proyecto, en la que se integra el repositorio de Vulhub, el archivo general `docker-compose.yml` y los scripts auxiliares de comprobación. El archivo de orquestación define los servicios vulnerables, sus nombres de contenedor, el mapeo de puertos, la política de reinicio y la red Docker compartida. Esta estructura levanta el entorno completo con un único comando y lo detiene después de forma controlada.

El flujo de despliegue consta de cuatro pasos: descargar Vulhub y seleccionar los entornos; adaptar el archivo de orquestación para integrarlos en una red Docker común; ejecutar Docker Compose para construir y levantar los contenedores; y ejecutar el script de validación para comprobar que los servicios están activos y accesibles desde el anfitrión.

El mapeo de puertos asocia cada vulnerabilidad a un puerto identificable, lo que simplifica la generación de tráfico y facilita el etiquetado posterior, al relacionar el tráfico observado con el servicio, la vulnerabilidad y el escenario experimental correspondiente.

La arquitectura es modular: para incorporar una nueva vulnerabilidad basta con añadir el servicio al archivo de despliegue, asignarle un puerto externo y conectarlo a la red Docker compartida; del mismo modo puede deshabilitarse cuando no se necesite. Así se construyen datasets por familias de ataque, tecnologías o escenarios de dificultad creciente, y se repiten experimentos en condiciones homogéneas.

2.4. *Validación operativa*

Una vez desplegado el entorno, se ejecuta un script de validación en PowerShell para comprobar el estado de Docker, contar los contenedores activos y verificar la accesibilidad de cada servicio. Para los servicios HTTP, el script realiza una petición mediante `Invoke-WebRequest`; si el servicio no responde correctamente a nivel HTTP, se aplica una comprobación TCP como mecanismo alternativo. Para los servicios no HTTP, como Redis, MySQL u OpenSSH, se utiliza directamente una prueba de conectividad TCP mediante `Test-NetConnection`.

La validación operativa inicial comprueba que los servicios vulnerables se despliegan correctamente, que sus puertos son accesibles desde el sistema anfitrión y que el entorno puede destruirse y recrearse de forma controlada mediante Docker Compose. Con múltiples contenedores, este paso distingue los errores de despliegue de los problemas del escenario experimental y verifica, antes de cada campaña, que el tráfico generado corresponde a una comunicación efectiva con el servicio objetivo.

2.5. *Generación y captura de tráfico*

El flujo experimental propuesto consta de cinco fases. En primer lugar, se selecciona uno de los servicios vulnerables desplegados en el MiniPC. En segundo lugar, se genera tráfico benigno o tráfico asociado a la vulnerabilidad correspondiente dentro de un entorno controlado. En tercer lugar, dicho tráfico se encamina a través del router Cradlepoint R1900 y de la red privada 5G. En cuarto lugar, el tráfico se captura desde el core 5G mediante `tcpdump`. Finalmente, las capturas obtenidas pueden almacenarse, documentarse y etiquetarse en función del servicio, vulnerabilidad, puerto, tipo de tráfico y fase del experimento.

Separar el punto de generación (extremo de aplicación) del de observación (core 5G) permite estudiar cómo aparece

el tráfico de los servicios vulnerables desde la infraestructura móvil y relacionar cada captura con un escenario controlado y una CVE documentada.

El tráfico capturado puede abarcar desde reconocimiento y enumeración hasta interacciones específicas con los servicios. No obstante, el objetivo no es documentar la explotación de cada vulnerabilidad, sino establecer la arquitectura que permite generar, capturar y organizar ese tráfico de forma reproducible; los detalles ofensivos quedan fuera del alcance. La Sección 4 materializa este flujo en una primera prueba de concepto.

3. Selección de servicios vulnerables

La selección de servicios vulnerables determina la diversidad del tráfico generado y, con ella, la utilidad de las capturas para construir datasets. No se busca reproducir un único ataque aislado, sino reunir un conjunto heterogéneo de servicios que genere tráfico asociado a distintas tecnologías, protocolos, puertos y familias de vulnerabilidades. Para ello se han seleccionado 19 entornos de Docker-Vulhub, vinculados a vulnerabilidades conocidas o a escenarios habituales en pruebas de penetración.

Los servicios responden a cuatro criterios: (i) vulnerabilidades documentadas mediante CVE o escenarios de explotación bien conocidos, de modo que cada captura tenga un contexto técnico claro; (ii) tecnologías habituales en entornos reales (frameworks web, gestores de contenido o CMS, servidores de aplicaciones, bases de datos, herramientas de administración y monitorización, y servicios de infraestructura); (iii) diversidad de familias de ataque (ejecución remota de código, inyección SQL, escritura arbitraria de archivos, path traversal, bypass de autenticación, deserialización y enumeración de usuarios); y (iv) estabilidad operativa para un despliegue simultáneo y reproducible.

Los servicios se agrupan en familias funcionales: ejecución remota de código (RCE), inyección y manipulación de parámetros, escritura arbitraria de archivos, acceso a servicios de datos y administración o reconocimiento. Esta agrupación muestra que el banco cubre superficies de ataque representativas sin reemplazar la descripción individual de cada CVE. Predominan las vulnerabilidades de RCE por su interés práctico: generan tráfico claramente diferenciable, tanto en la interacción con el servicio como en las comunicaciones posteriores de escenarios más avanzados.

Desde el punto de vista experimental, esta variedad genera capturas heterogéneas: los servicios HTTP producen tráfico web (parámetros, cabeceras y rutas modificables), mientras que los tres servicios TCP no HTTP (Redis, MySQL y OpenSSH) aportan patrones de conexión propios de infraestructura o administración, lo que evita que el dataset quede sesgado hacia una única tecnología. Asimismo, la selección reproduce fases habituales de un escenario controlado: reconocimiento y enumeración (OpenSSH), bypass de autenticación (MySQL), inyección y manipulación de parámetros (Joomla, phpMyAdmin) y ejecución remota de código o deserialización (WordPress, Drupal, WebLogic, Spring, Struts2, Solr, Fastjson).

Esto proporciona una base flexible para campañas progresivas: una primera campaña puede generar tráfico benigno como línea base y las siguientes activar interacciones asociadas a vulnerabilidades concretas, etiquetando cada captura por servicio, CVE, puerto, familia de ataque y fase del experimento.

3.1. Entorno de servicios vulnerables

La versión inicial del banco de pruebas incluye 19 servicios vulnerables desplegados de forma simultánea sobre la red Docker común descrita en la Sección 2, cada uno publicado en un puerto externo distinto. No se cubren exhaustivamente todas las familias de vulnerabilidades, sino que se reúne un conjunto heterogéneo y operativo, priorizando la estabilidad de despliegue, la disponibilidad en Docker-Vulhub y la capacidad de generar tráfico diferenciable a nivel de red.

La Tabla 1 resume los servicios incluidos, que abarcan aplicaciones web, CMS, bases de datos, administración, monitorización e infraestructura, y cubren distintas familias de vulnerabilidades.

Los contenedores aíslan cada servicio vulnerable y facilitan destruir y reconstruir el entorno cuando sea necesario. Algunos servicios se despliegan a partir de imágenes ya disponibles, mientras que otros requieren construir la imagen localmente a partir de los ficheros proporcionados por Vulhub. Esta característica es relevante, ya que Vulhub no funciona únicamente como un repositorio de imágenes preconstruidas, sino como una colección de entornos reproducibles que en muchos casos deben generarse localmente mediante los correspondientes archivos `Dockerfile` y `docker-compose.yml`.

4. Prueba de concepto: captura desde el core 5G

Para validar la viabilidad del banco de pruebas se desplegó el entorno y se capturó, desde el core 5G SA, un escenario de ataque real contra uno de los servicios: Apache Tomcat (CVE-2017-12615), que admite la escritura arbitraria de un archivo `.jsp` y la consiguiente ejecución remota de código. La captura, tomada sobre la infraestructura OAI del Firecell Labkit 40, contiene 7616 tramas a lo largo de 252 s (8,2 MB). No pretende ser exhaustiva, sino mostrar que un ataque a un servicio vulnerable es observable desde el core y que de la captura puede derivarse un conjunto de datos etiquetado.

La captura recoge a la vez los dos planos de la red 5G: el plano de usuario circula por N3 tunelizado mediante GTP-U (UDP 2152) entre el gNB y la UPF (7522 tramas GTP-U), y el plano de control aparece en N2 como señalización NGAP sobre SCTP (puerto de servicio 38412 del AMF) entre el AMF y el gNB (64 tramas). El atacante, en la red de datos, alcanza el servicio alojado en el equipo conectado como terminal 5G (UE); dicho servicio, publicado mediante mapeo de puertos, se observa de-encapsulado en la dirección del UE sobre un puerto traducido por NAT, efecto que también condiciona el etiquetado (Sección 5).

Tras de-encapsular el túnel GTP-U se reconstruye la secuencia completa del ataque: una petición `PUT /shell.jsp/` que recibe `201 Created` (escritura del *webshell*), seguida de `GET /shell.jsp` (`200 OK`) y de sucesivas peticiones `GET /shell.jsp?cmd=...` (`whoami`, `ls`, `dir`) cuyas respuestas devuelven la salida de los comandos (por ejemplo, el listado del directorio de Tomcat), confirmando la ejecución remota de código. El ataque, automatizado con `python-requests`, se concreta en 9 conexiones TCP, 196 paquetes y unos 37 KB.

Tabla 1: Servicios vulnerables incluidos en la versión inicial del banco de pruebas.

Servicio	Vulnerabilidad	Puerto externo	Tipo	Tecnología
Spring Cloud Gateway	CVE-2022-22947	8180	RCE	Web/API
Apache Struts2	CVE-2017-5638	8081	RCE	Web
Drupal	CVE-2019-6339	8082	RCE	CMS
Jenkins	CVE-2018-1000861	8083	RCE	DevOps
Redis	CVE-2022-0543	6379	RCE	Base de datos
Apache Druid	CVE-2021-25646	8888	RCE	Analítica
PHP-FPM	CVE-2019-11043	8086	RCE	Web
Apache Tomcat	CVE-2017-12615	8087	Escritura arbitraria/RCE	Web
Elasticsearch	CVE-2014-3120	9200	RCE	Búsqueda
MySQL	CVE-2012-2122	3307	Bypass de autenticación	Base de datos
phpMyAdmin	CVE-2018-12613	8088	LFI/RCE	Administración
WordPress	PwnScriptum	8089	RCE	CMS
Nexus	CVE-2019-7238	8090	RCE	Repositorio
Grafana	CVE-2021-43798	3000	Path traversal	Monitorización
OpenSSH	CVE-2018-15473	2222	Enumeración	Administración
Joomla	CVE-2017-8917	8091	Inyección SQL	CMS
WebLogic	CVE-2017-10271	7001	RCE	Middleware
Apache Solr	CVE-2017-12629	8983	RCE	Búsqueda
Fastjson	1.2.24-rce	8092	RCE	Java/JSON

Un ataque de capa de aplicación atraviesa así el plano de usuario 5G y queda analizable desde el core.

A partir del tráfico de-encapsulado se construyó un pequeño conjunto de datos etiquetado, a nivel de paquete y de flujo, con tres clases: el ataque, el tráfico benigno del UE (QUIC, TLS y DNS hacia 21 destinos públicos: actualizaciones del sistema, telemetría, CDN y DNS) y la señalización de control N2. La Tabla 2 resume su composición y la Figura 2, su coexistencia temporal en una única captura. El etiquetado se ancla en el contenido observado (la firma del ataque), lo que lo hace reproducible; el script de generación y el conjunto de datos resultante están disponibles en abierto.[1]

Tabla 2: Composición del tráfico capturado desde el core 5G y etiquetado (escenario CVE-2017-12615).

Clase	Plano (interfaz)	Paq.	Flujos
Ataque Tomcat (RCE)	Usuario (N3)	196	9
Tráfico benigno	Usuario (N3)	2350	63
Señalización (NGAP)	Control (N2)	64	1

Además, 4976 fragmentos IP del túnel GTP-U (plano de usuario) y 30 tramas internas del core (*loopback*) no se incluyen como muestras del dataset; total: 7616 tramas.

5. Discusión

El banco de pruebas se diferencia de los conjuntos previos en dos aspectos. Frente a los datasets generados sobre infraestructuras no 5G (KDD, UNSW-NB15, CICIDS2017 o Bot-IoT) o a los escenarios simulados, hace circular el tráfico por una red privada 5G SA real y lo observa desde el core, relacionando el tráfico de aplicación con su correlato en la señalización y los túneles del core 5G. Frente a las capturas aisladas, organizar los servicios por contenedor, puerto y familia facilita un etiquetado trazable de cada captura.

El enfoque presenta limitaciones. En primer lugar, más allá de la prueba de concepto descrita (limitada a un único servicio y a una captura corta), la validación sistemática sobre todos los servicios y la publicación de un conjunto de datos completo se plantean como trabajo futuro. En segundo lugar, los servicios contenerizados no reproducen el comportamiento de dispositivos OT/IoT físicos, por lo que el tráfico generado es una aproximación reproducible, no un sustituto completo de equipos reales. En tercer lugar, la publicación de los servicios mediante mapeo de puertos en el equipo anfitrión introduce traducción de direcciones y puertos (NAT), lo que altera parte de la información de red observada desde el núcleo y debe tenerse en cuenta durante el etiquetado de las trazas.

Por último, el punto de observación condiciona qué resulta capturable. Como se ha mostrado en la Sección 4, el tráfico de usuario viaja tunelizado mediante GTP-U por N3 (capturable en la UPF, en N3 o ya sin túnel en la interfaz N6 de salida) y la señalización de control aparece como NGAP sobre SCTP en N2. La elección del punto de captura determina qué características son observables y, en consecuencia, la utilidad del conjunto de datos para entrenar y evaluar sistemas IDS.

6. Conclusiones y trabajo futuro

Este trabajo ha presentado una arquitectura experimental para la generación controlada de tráfico de ciberseguridad en una red privada 5G. La propuesta combina servicios vulnerables contenerizados, basados en Docker-Vulhub, con una infraestructura Firecell Labkit 40 en modo 5G SA. El objetivo no ha sido evaluar de forma exhaustiva cada vulnerabilidad, sino definir y documentar un banco de pruebas modular que permita desplegar servicios con CVE conocidas, generar tráfico asociado a distintas familias de ataque y capturarlo desde el core de la red 5G, como se ha validado en una prueba de concepto sobre Apache Tomcat (CVE-2017-12615).

La arquitectura reduce algunas barreras habituales en la creación de datasets de ciberseguridad en 5G: los contenedores facilitan reproducir, destruir y reconstruir el entorno sin depender de dispositivos físicos, y la integración con la red privada 5G permite observar el tráfico desde una perspectiva próxima a la infraestructura móvil real, relevante para supervisar la seguridad de los lazos de control y automatización que dependen de la red 5G privada. La versión inicial incluye 19 servicios heterogéneos (ejecución remota de código, inyección SQL, escritura arbitraria de archivos, path traversal, bypass de autenticación, deserialización y enumeración), ampliables o sustituibles sin modificar la arquitectura 5G subyacente.

[1] `https://github.com/joseAveleira/5g-vulhub-dataset`

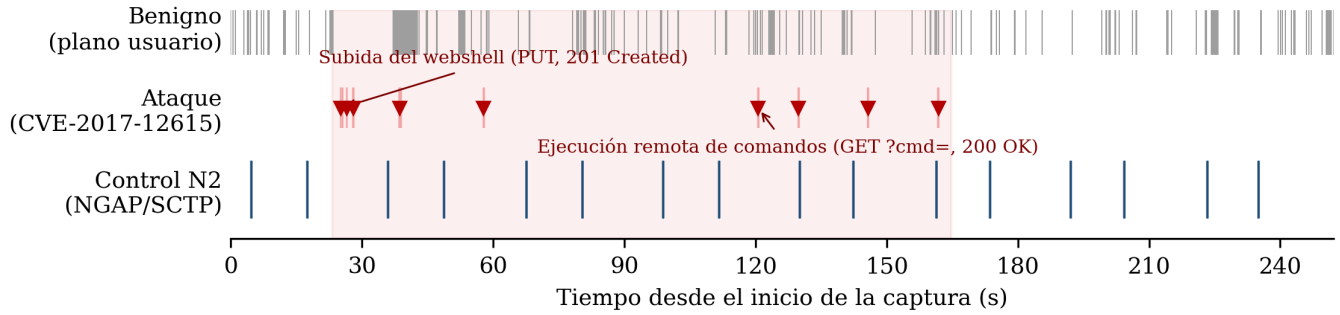

Figura 2: Cronología del tráfico capturado desde el core 5G. El ataque a Tomcat (CVE-2017-12615), reconstruido tras de-encapsular GTP-U, coexiste con el tráfico benigno del plano de usuario y con la señalización de control N2 (NGAP/SCTP) en una sola captura.

Como trabajo futuro se plantea utilizar esta arquitectura para campañas sistemáticas de captura en las que cada traza quede asociada al servicio, la vulnerabilidad, el puerto, la familia de ataque y la fase del experimento, construyendo datasets etiquetados que diferencien tráfico normal y malicioso. Se prevé ampliar el banco con nuevos servicios, automatizar parcialmente la generación de tráfico, incorporar perfiles benignos más realistas y evaluar modelos de IDS y aprendizaje automático sobre los datos generados, analizando qué características observables desde el core resultan más discriminantes. Finalmente, los artefactos de despliegue, el conjunto de datos y el código de generación se han publicado en un repositorio abierto, lo que facilita la reproducción y adaptación del entorno por parte de otros laboratorios con infraestructuras 5G privadas o equivalentes.

Referencias

Alsaedi, A., Moustafa, N., Tari, Z., Mahmood, A., Anwar, A., 2020. TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems. IEEE Access 8, 165130–165150.
DOI: 10.1109/ACCESS.2020.3022862

Amponis, G., Radoglou-Grammatikis, P., Lagkas, T., Mallouli, W., Cavalli, A., Klonidis, D., Markakis, E., Sarigiannidis, P., 12 2022. Threatening the 5g core via pfcp dos attacks: the case of blocking uav communications. Eurasip Journal on Wireless Communications and Networking 2022.
DOI: 10.1186/S13638-022-02204-5

Coldwell, C., Conger, D., Goodell, E., Jacobson, B., Petersen, B., Spencer, D., Anderson, M., Sgambati, M., 2022. Machine learning 5g attack detection in programmable logic. 2022 IEEE GLOBECOM Workshops, GC Wkshps 2022 - Proceedings, 1365–1370.
DOI: 10.1109/GCWKSHPS56602.2022.10008647

Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L., Janicke, H., 2022. Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. IEEE Access 10, 40281–40306.
DOI: 10.1109/ACCESS.2022.3165809

Goldschmidt, P., Chudá, D., 2025. Network intrusion datasets: A survey, limitations, and recommendations. Computers & Security 156, 104510.
DOI: 10.1016/j.cose.2025.104510

Hamroun, C., Fladenmuller, A., Pariente, M., Pujolle, G., 2025. Intrusion detection in 5g and wi-fi networks: A survey of current methods, challenges, and perspectives. IEEE Access 13, 40950–40976.
DOI: 10.1109/ACCESS.2025.3546338

Khan, M. S., Farzaneh, B., Shahriar, N., Saha, N., Boutaba, R., 2022. Slice-secure: Impact and detection of dos/ddos attacks on 5g network slices. In: 2022 IEEE Future Networks World Forum (FNWF). pp. 639–642.
DOI: 10.1109/FNWF55208.2022.00117

Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B., 2019. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. Future Generation Computer Systems 100, 779–796.
DOI: 10.1016/j.future.2019.05.041

Moustafa, N., Slay, J., 2015. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 Military Communications and Information Systems Conference (MilCIS). pp. 1–6.
DOI: 10.1109/MilCIS.2015.7348942

Narciandi-Rodríguez, D., Martínez-Martínez, G., Aveleira-Mata, J., Bayón Gutiérrez, M., Alfonso-Cendón, J., García-Rodríguez, I., 2026. Design and capture of a 5g sa traffic dataset under jamming conditions. In: Rojas, I., Joya, G., Catala, A. (Eds.), Advances in Computational Intelligence. Springer Nature Switzerland, Cham, pp. 261–273.

Narciandi-Rodríguez, D., Aveleira-Mata, J., García-Ordás, M. T., Alfonso-Cendón, J., Benavides, C., Alaiz-Moretón, H., 2025a. A cybersecurity review in iot 5g networks. Internet of Things 30, 101478.
DOI: 10.1016/j.iot.2024.101478

Narciandi-Rodríguez, D., Martínez-Martínez, G., Aveleira-Mata, J., Bayón Gutiérrez, M., Alfonso-Cendón, J., García-Rodríguez, I., 2025b. 5G-RaaS-UAD 5G Ransomware as a Service Attack – Under Attack Dataset.
URL: https://figshare.com/s/160750e5d36974bb4c71?file=59439863

Noor, K., Imoize, A. L., Li, C.-T., Weng, C.-Y., 2025. A review of machine learning and transfer learning strategies for intrusion detection systems in 5g and beyond. Mathematics 13 (7), 1088.
DOI: 10.3390/math13071088

Sharafaldin, I., Lashkari, A. H., Ghorbani, A. A., 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP). pp. 108–116.
DOI: 10.5220/0006639801080116

Siriwardhana, Y., Samarakoon, S., Porambage, P., Liyanage, M., Chang, S.-Y., Kim, J., Kim, J., Ylianttila, M., 2025. Descriptor: 5g wireless network intrusion detection dataset (5g-nidd). IEEE Data Descriptions 2, 11098458.
DOI: 10.1109/IEEEDATA.2025.3592888

Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. A., 2009. A detailed analysis of the kdd cup 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications. pp. 1–6.
DOI: 10.1109/CISDA.2009.5356528

Thakkar, A., Lohiya, R., 2020. A review of the advancement in intrusion detection datasets. Procedia Computer Science 167, 636–645.
DOI: 10.1016/j.procs.2020.03.330

Vulhub, 2026. Vulhub - open-source vulnerable docker environments.
URL: https://vulhub.org/