B28

BUSSIO28TEAM

GUÍA COMPLETA DE METASPLOIT

PENTESTING PROFESIONAL Y ÉTICO

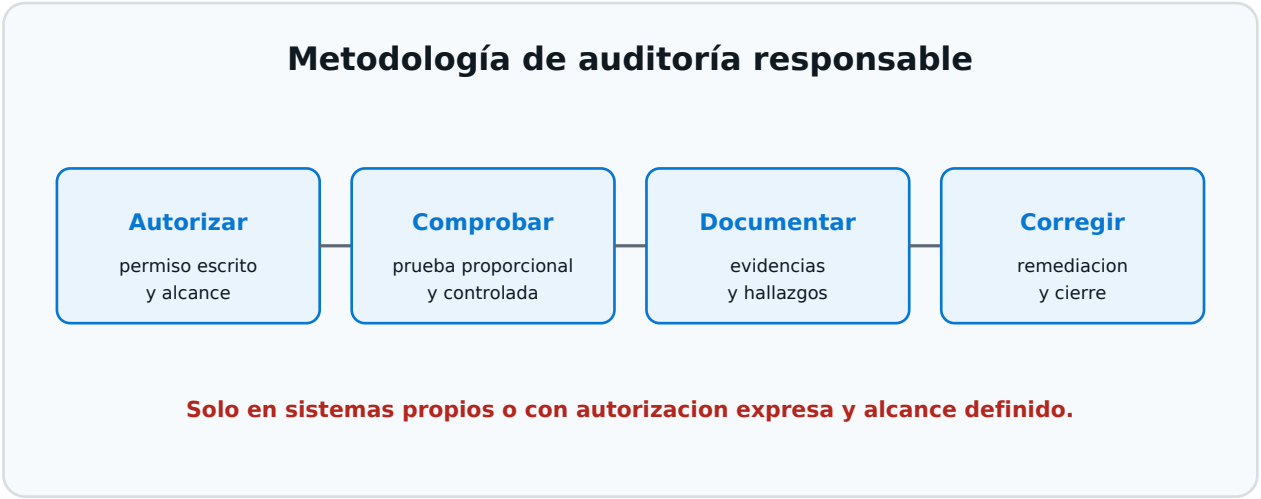
Desde la instalación hasta la explotación, post-explotación y elaboración de informes

by Antoni Clarens

ALMERÍA, ESPAÑA

GUÍA PRÁCTICA DE METASPLOIT FRAMEWORK

Edición con marco legal, propiedad intelectual y uso responsable



Autor: Antoni Clarens

Bussio28Team · España · Edición 2026

Página legal y condiciones de uso

Titularidad de la obra

El texto, estructura pedagógica, selección y organización de contenidos, explicaciones, diagramas y presentación de esta guía constituyen una obra elaborada para Bussio28Team y atribuida a Antoni Clarens. Salvo indicación expresa, no se concede licencia para reproducir, transformar, distribuir, vender, comunicar públicamente o incorporar de forma sustancial esta obra a otra publicación. Las citas breves y demás usos permitidos por la legislación aplicable quedan sujetos a sus requisitos legales.

Marcas, software y materiales de terceros

Metasploit Framework es un proyecto de código abierto mantenido por Rapid7 y distribuido principalmente bajo licencia BSD de tres cláusulas, con componentes de terceros que pueden estar sujetos a licencias distintas. Esta guía no redistribuye el software ni pretende sustituir sus licencias. Los nombres Metasploit, Metasploit Framework, Rapid7 y los nombres de productos o proyectos citados se utilizan únicamente con finalidad identificativa y formativa; los derechos correspondientes pertenecen a sus titulares.

Independencia editorial

Bussio28Team y Antoni Clarens no afirman relación comercial, patrocinio, certificación, autorización ni afiliación con Rapid7 por la mera publicación de esta guía. La documentación oficial debe considerarse la referencia para versiones, licencias y comportamiento actual del software.

Finalidad y autorización

Esta guía está destinada a formación, investigación defensiva, laboratorios propios, CTF autorizados y auditorías de seguridad con permiso previo. El lector es responsable de disponer de autorización suficiente antes de interactuar con un sistema, servicio, cuenta o red que no sea de su exclusiva propiedad. La existencia de una vulnerabilidad, la accesibilidad de un servicio o la disponibilidad pública de una herramienta no constituyen por sí mismas autorización para probarlo.

Marco penal español

En España, determinadas conductas de acceso o permanencia no autorizados en sistemas de información pueden tener consecuencias penales. Por ello, los ejercicios de esta obra deben limitarse a entornos controlados o a encargos profesionales cuyo alcance haya sido expresamente autorizado. Cuando una auditoría se realice para terceros, es recomendable conservar la autorización, el alcance, las fechas, los activos incluidos, las exclusiones y un contacto de emergencia.

Protección de datos y confidencialidad

Una auditoría puede exponer nombres de usuario, direcciones IP, registros, documentos, credenciales, datos personales o información empresarial. Debe aplicarse minimización: recopilar solo lo necesario, proteger las evidencias, limitar el acceso, definir conservación y eliminación, y respetar los acuerdos de confidencialidad y la normativa de protección de datos aplicable. Las capturas destinadas a informes o formación deben anonimizar información innecesaria.

Exención de garantías y responsabilidad

La guía tiene finalidad educativa y no constituye asesoramiento jurídico ni garantiza que un procedimiento técnico sea adecuado para todos los entornos. El software y sus módulos cambian con el tiempo. El usuario debe revisar la versión instalada, la documentación oficial y las condiciones de su autorización antes de ejecutar una prueba. Ni la autoría de esta guía ni Bussio28Team autorizan mediante este documento el acceso a sistemas de terceros.

Versión y conservación

Edición 1.1 - 2026. Para acreditar una edición concreta conviene conservar el PDF original, su fecha de publicación y, si se distribuye públicamente, un hash criptográfico del archivo. Si se comercializa o publica

como libro, puede añadirse posteriormente ISBN, depósito legal u otros datos editoriales cuando proceda.

PRÓLOGO

Comprender Metasploit para comprender la seguridad ofensiva

Aprender Metasploit no consiste en memorizar comandos ni en ejecutar módulos de forma mecánica. Consiste en aprender a formular preguntas técnicas, interpretar la información que devuelve un sistema y relacionar una vulnerabilidad con su contexto real. La herramienta cobra sentido cuando se utiliza como parte de una metodología: reconocimiento, análisis, validación, evaluación del impacto, documentación y corrección.

Para un hacker ético, conocer el funcionamiento de Metasploit es prácticamente imprescindible. No porque sea la única herramienta de seguridad ofensiva, sino porque reúne en un mismo entorno muchos de los conceptos fundamentales del pentesting moderno: exploits, payloads, módulos auxiliares, sesiones, post-explotación, opciones de red, arquitectura del objetivo y evidencias reproducibles. Dominar estos conceptos permite después comprender con mayor facilidad otras herramientas y, sobre todo, saber qué está ocurriendo realmente detrás de cada prueba.

La diferencia entre utilizar una herramienta y comprenderla es decisiva. Un profesional no debería limitarse a preguntar «qué módulo funciona», sino también «por qué funciona», «qué condición vulnerable está validando», «qué tráfico genera», «qué privilegios puede obtener», «qué riesgo introduce la prueba» y «cómo puede corregirse el problema». Esa forma de razonar convierte una demostración técnica en una auditoría útil.

Metasploit debe entenderse como un laboratorio de conceptos de seguridad. Permite observar de forma estructurada cómo una debilidad puede pasar de ser una referencia técnica o un CVE a convertirse en una evidencia verificable. Al mismo tiempo, enseña una lección igual de importante: que no toda vulnerabilidad debe explotarse para demostrar su existencia y que el buen auditor busca siempre la prueba suficiente con el menor impacto posible.

El conocimiento técnico solo adquiere valor profesional cuando se acompaña de criterio, autorización y responsabilidad.

Esta guía está concebida para avanzar desde los fundamentos hasta escenarios prácticos controlados. El lector encontrará arquitectura de módulos, uso de msfconsole, reconocimiento, sesiones, metodología de auditoría y ejemplos orientados a comprender tanto la explotación como la detección y mitigación.

PRÓLOGO

De la herramienta a la metodología

Conocer Metasploit también ayuda a pensar como defensor. Quien entiende cómo se selecciona un objetivo, cómo se identifica una versión vulnerable y qué condiciones necesita un exploit puede diseñar mejores controles, segmentar servicios, priorizar parches y reconocer señales de actividad anómala. Seguridad ofensiva y defensa no son disciplinas aisladas: comparten el mismo conocimiento técnico visto desde perspectivas complementarias.

Por ello, el objetivo de estas páginas no es presentar Metasploit como un botón automático para «hackear» sistemas. La intención es exactamente la contraria: enseñar a trabajar con orden. Antes de cualquier prueba deben existir un alcance definido, autorización expresa, un entorno adecuado y una estrategia de recuperación. Durante la prueba deben registrarse las decisiones y limitarse las acciones a lo necesario. Al terminar, deben cerrarse sesiones, eliminarse artefactos y documentarse las medidas correctoras.

El laboratorio con máquinas deliberadamente vulnerables, como Metasploitable, es especialmente valioso porque permite equivocarse, repetir y comparar resultados sin trasladar ese riesgo a sistemas reales. Es el lugar apropiado para aprender qué significa configurar RHOSTS, seleccionar un módulo, interpretar sus opciones o comprender por qué un servicio concreto resulta vulnerable.

A medida que aumenta la experiencia, Metasploit deja de parecer una colección de comandos y empieza a verse como un marco de trabajo. Ese cambio de perspectiva es uno de los objetivos centrales de la guía. El lector debería terminar cada ejercicio siendo capaz de explicar qué ha hecho, por qué lo ha hecho, qué evidencia ha obtenido y cómo mitigaría el hallazgo.

La excelencia en hacking ético no se mide por la cantidad de sistemas comprometidos, sino por la capacidad de identificar riesgos con precisión, demostrar su impacto de forma controlada y ayudar a corregirlos.

Esta obra debe utilizarse exclusivamente en sistemas propios, laboratorios, CTF o infraestructuras para las que exista autorización expresa. La técnica sin ética es incompleta; la ética sin conocimiento técnico no basta para auditar con rigor. La combinación de ambas es la base del trabajo profesional.

Antoni Clarens
Bussio28Team · Almería, España · 2026

Índice

1. Fundamentos y alcance

Metasploit Framework es una plataforma modular utilizada para organizar comprobaciones de seguridad. Su utilidad profesional aparece cuando se integra en un proceso de autorización, reconocimiento, análisis, validación proporcional y documentación. Un resultado técnico nunca sustituye el juicio del auditor.

2. Preparación del laboratorio

Para aprender, utiliza máquinas virtuales aisladas, snapshots y objetivos deliberadamente vulnerables. Evita puentes de red innecesarios y no reutilices credenciales reales. Documenta las direcciones internas y restaura las máquinas al finalizar.

3. msfconsole

La interfaz principal es msfconsole. La ayuda integrada permite descubrir comandos y revisar el contexto activo.

```
msfconsole
help
version
search
info
use
show options
back
exit
```

4. Arquitectura de módulos

Las familias más conocidas son auxiliary, exploit, payload y post. Auxiliary cubre tareas de apoyo y reconocimiento; exploit representa lógica para validar vulnerabilidades; payload define acciones asociadas a determinados flujos; post reúne acciones posteriores sobre una sesión autorizada.

5. Búsqueda y selección

Busca por producto, servicio, plataforma o CVE y lee la información del módulo antes de seleccionarlo. Comprueba referencias, requisitos y compatibilidad.

```
search type:auxiliary http
search name:smb
search cve:AAAA-NNNN
info <modulo>
use <modulo>
```

6. Opciones y variables

show options presenta los parámetros. RHOSTS suele identificar objetivos remotos y RPORT un puerto remoto; LHOST/LPORT aparecen en determinados flujos. No todos los módulos usan las mismas variables.

```
show options
show advanced
set RHOSTS <IP_DE_LABORATORIO>
```

```
unset RHOSTS
```

7. Reconocimiento y auxiliary

Empieza por enumeración y reconocimiento de bajo impacto. La finalidad es confirmar qué servicios existen y construir contexto antes de plantear cualquier validación.

8. Exploit y payload: conceptos

Un exploit intenta demostrar un efecto de una vulnerabilidad; un payload puede definir la acción posterior. En una auditoría real debe preferirse check u otra comprobación no destructiva cuando esté disponible.

9. Sesiones y Meterpreter

Una sesión representa un canal activo obtenido durante una prueba. Meterpreter es un tipo de sesión con funciones de interacción. Esta guía limita su uso a administración básica de sesiones y evita técnicas de persistencia, evasión o recopilación de credenciales.

10. Base de datos y workspaces

Los workspaces permiten separar proyectos y asociar hosts, servicios, notas y hallazgos al alcance correcto.

```
workspace
workspace -a laboratorio
hosts
services
vulns
notes
```

11. Importación de reconocimiento

La importación permite centralizar resultados de herramientas compatibles. Importa únicamente datos obtenidos dentro del alcance autorizado.

12. Jobs y sesiones

Los jobs son tareas en segundo plano. Revisa y detén trabajos que ya no sean necesarios; al cerrar una auditoría no deben quedar listeners o procesos olvidados.

13. Msfvenom: introducción segura

msfvenom permite trabajar con payloads y formatos. En una guía inicial es suficiente conocer su ayuda y capacidades. La generación de artefactos ejecutables debe reservarse a laboratorios aislados y finalidades expresamente autorizadas.

```
msfvenom --help
msfvenom -l payloads
msfvenom -l formats
```

14. Módulos post

Los módulos post actúan sobre sesiones existentes. Esta fase requiere especial cuidado porque puede acceder a información sensible y exceder fácilmente el objetivo inicial.

15. Automatización básica

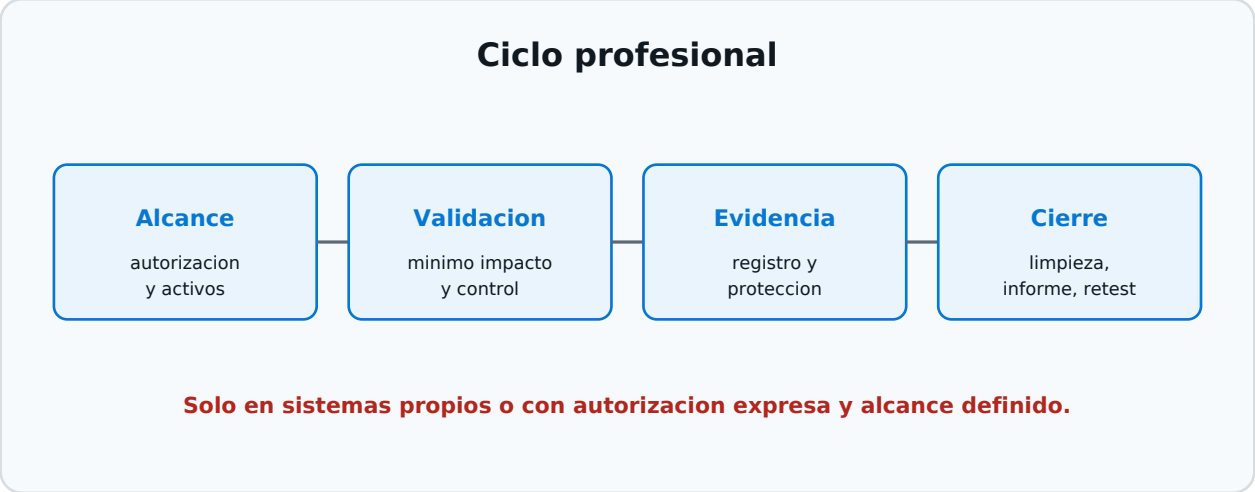
Los resource scripts ayudan a reproducir secuencias de consola. Una automatización profesional debe ser revisable, acotada y libre de acciones destructivas no necesarias.

16. Actualización y diagnóstico

Revisa version, help, info y show options. Los módulos cambian; nunca asumas que una instrucción antigua conserva exactamente el mismo comportamiento.

17. Metodología de auditoría

Secuencia recomendada: autorización; alcance; reconocimiento; hipótesis; selección del módulo; lectura técnica; comprobación de bajo impacto; validación autorizada; evidencias; limpieza; informe; remediación y retest.



18. Laboratorio guiado

Crea un workspace, identifica una VM de práctica, utiliza un módulo auxiliary apropiado para observar un servicio, registra el resultado y cierra el entorno. El ejercicio enseña el ciclo sin necesitar una explotación real.

19. Evidencias e informe

Registra activo, fecha, herramienta y versión, hallazgo, evidencia, impacto, pasos resumidos, recomendación y resultado del retest. Protege y anonimiza datos sensibles.

20. Ética, legalidad y reglas de compromiso

Antes de comenzar una auditoría de terceros debe existir un documento de reglas de compromiso. Debe identificar quién autoriza, activos incluidos, activos excluidos, horarios, técnicas permitidas, límites sobre datos, contactos, criterios de parada, gestión de evidencias y procedimiento de cierre. Si el alcance es dudoso, la prueba se detiene hasta aclararlo.

21. Glosario y comandos

Comandos esenciales: help, search, info, use, show options, set, unset, check, run, back, sessions, jobs, workspace, hosts, services e history. La ayuda de la versión instalada prevalece sobre cualquier chuleta.

22. Recursos y referencias

Referencia técnica: documentación oficial de Metasploit Framework y repositorio oficial de Rapid7. Referencia jurídica española: legislación vigente publicada en el BOE. Las marcas y licencias de componentes de terceros deben respetarse según corresponda.

Anexo A. Modelo de autorización de laboratorio / auditoría

Este modelo es orientativo y debe adaptarse al contrato y al asesoramiento jurídico aplicable. No sustituye un documento profesional de reglas de compromiso.

Entidad o propietario que autoriza: _____
Auditor / equipo autorizado: _____
Fecha y ventana de pruebas: _____
Activos incluidos (IP, dominios, aplicaciones): _____
Activos expresamente excluidos: _____
Técnicas autorizadas: _____
Técnicas prohibidas / límites de impacto: _____
Tratamiento y conservación de evidencias: _____
Contacto de emergencia: _____
Criterio de parada inmediata: _____
Autorización expresa para las pruebas descritas: SI / NO

Firma de quien autoriza: _____ Fecha: _____
Firma del auditor: _____ Fecha: _____

Una autorización genérica puede no ser suficiente para técnicas de alto impacto. El alcance debe ser concreto y comprensible para ambas partes.

Anexo B. Checklist legal y técnico previo

Antes de ejecutar una prueba confirma:

1. Existe autorización identificable y vigente.
2. El activo está dentro del alcance.
3. La fecha y horario están permitidos.
4. La técnica prevista no está excluida.
5. Se ha definido un contacto de emergencia.
6. Existe copia de seguridad o plan de recuperación cuando proceda.
7. Se ha elegido la comprobación de menor impacto posible.
8. Está definido qué evidencias pueden conservarse.
9. Los datos personales o secretos se minimizarán y protegerán.
10. Se conoce el criterio de parada.
11. Se registrará la versión de la herramienta y el módulo utilizado.
12. Se cerrarán sesiones, jobs y recursos temporales al finalizar.
13. Se redactará un informe y se propondrá remediación.
14. Se realizará retest solo si está autorizado.

Nota jurídica final

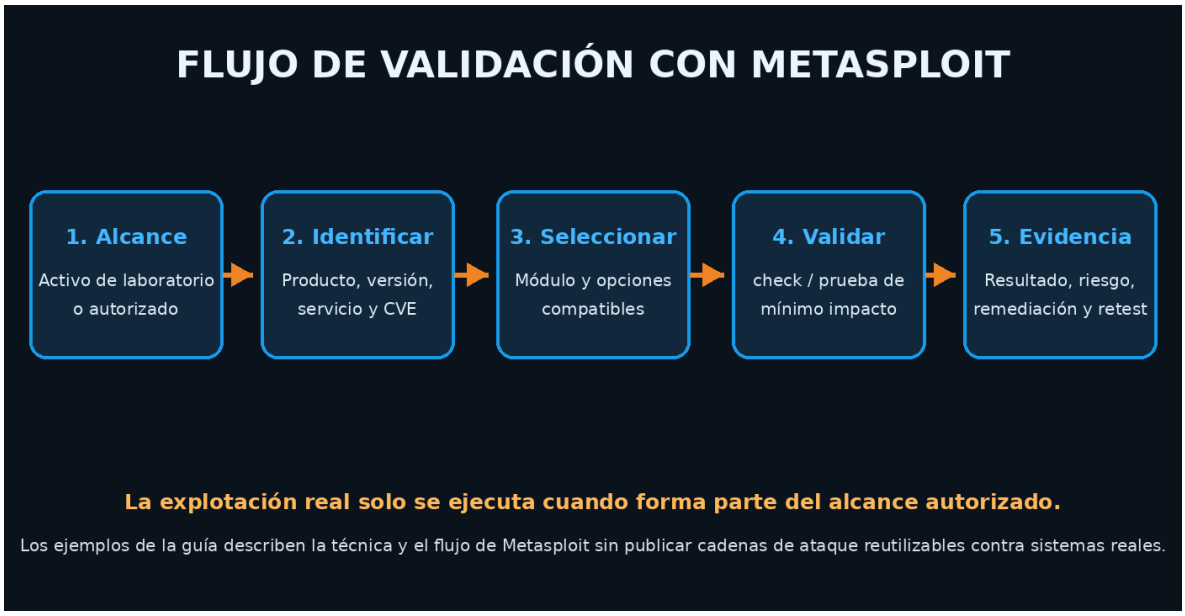
La inclusión de avisos legales no convierte por sí sola una actividad técnica en lícita. La licitud depende, entre otros factores, de la autorización, el alcance, la conducta realizada, los datos tratados y la normativa aplicable. Para publicación comercial, contratación de auditorías o casos dudosos, conviene revisión por un profesional jurídico.

TÉCNICA DE METASPLOIT: EJEMPLOS DE EXPLOTACIÓN

Esta sección incorpora cinco casos recientes de vulnerabilidades que han sido explotadas activamente. Se utilizan como estudios de caso para aprender a razonar con Metasploit: identificar la superficie, localizar un módulo compatible, revisar sus requisitos, validar de forma controlada y documentar el resultado.

IMPORTANTE

Los casos no incluyen instrucciones operativas completas, payloads funcionales ni cadenas de explotación contra productos reales. La práctica debe hacerse con máquinas de laboratorio expresamente preparadas para ello.



CASO 1 - Microsoft SharePoint Server

CVE-2026-58644

Qué ocurrió

Deserialización de datos no confiables con impacto de ejecución remota. CISA registró explotación activa y NIST/CISA la calificaron como automatizable y de impacto técnico total.

Flujo de Metasploit en laboratorio

Familia Metasploit: exploit remoto de aplicación web. En laboratorio, el auditor buscaría por CVE/producto, revisaría 'info', 'show options' y 'show targets', configuraría únicamente la VM vulnerable y usaría 'check' si el módulo lo soporta.

Qué hace diferente este ejemplo

Diferencia técnica: flujo web de deserialización. La evidencia útil es confirmar versión vulnerable, respuesta del módulo de comprobación y versión corregida tras el retest.

Defensa y cierre

Prioridad: aplicar las actualizaciones del fabricante, restringir exposición de SharePoint y revisar indicios de compromiso.

SECUENCIA DE CONSOLA - PLANTILLA SEGURA

```
search cve:<CVE>
info <modulo>
use <modulo>
show options
set RHOSTS <IP_VM_LAB>
check   # si está soportado
back
```

CASO 2 - Fortinet FortiSandbox

CVE-2026-25089

Qué ocurrió

Inyección de comandos del sistema operativo en FortiSandbox. CISA la añadió al catálogo KEV por evidencia de explotación activa.

Flujo de Metasploit en laboratorio

Familia Metasploit: exploit de command injection sobre appliance. El ejercicio de laboratorio debe limitarse a un clon/VM autorizado; se selecciona el módulo correspondiente, se revisan opciones HTTP/HTTPS y se valida primero la presencia de la condición vulnerable.

Qué hace diferente este ejemplo

Diferencia técnica: no depende de deserialización; el fallo está en el tratamiento de datos que terminan alcanzando una operación del sistema.

Defensa y cierre

Prioridad: parchear o aplicar mitigaciones del fabricante, limitar administración y revisar registros del appliance.

SECUENCIA DE CONSOLA - PLANTILLA SEGURA

```
search cve:<CVE>
info <modulo>
use <modulo>
show options
set RHOSTS <IP_VM_LAB>
check   # si está soportado
back
```

CASO 3 - SonicWall SMA1000

CVE-2026-15410

Qué ocurrió

Vulnerabilidad de inyección de código en appliances SMA1000. CISA la incluyó entre vulnerabilidades conocidas como explotadas.

Flujo de Metasploit en laboratorio

Familia Metasploit: exploit remoto de appliance/VPN. En una reproducción autorizada se comprobarían versión y endpoint, se elegiría el target apropiado y se utilizaría una acción inocua o 'check' cuando exista.

Qué hace diferente este ejemplo

Diferencia técnica: superficie de acceso remoto perimetral. El riesgo operativo es mayor porque estos equipos suelen estar expuestos a Internet; por eso la guía no publica una receta reutilizable.

Defensa y cierre

Prioridad: actualización inmediata, reducción de exposición y búsqueda de actividad anómala.

SECUENCIA DE CONSOLA - PLANTILLA SEGURA

```
search cve:<CVE>
info <modulo>
use <modulo>
show options
set RHOSTS <IP_VM_LAB>
check   # si está soportado
back
```

CASO 4 - Progress LoadMaster

CVE-2026-8037

Qué ocurrió

Inyección de comandos en Progress LoadMaster. CISA la añadió al KEV el 7 de agosto de 2026 por explotación conocida.

Flujo de Metasploit en laboratorio

Familia Metasploit: exploit remoto contra dispositivo de infraestructura. El flujo formativo consiste en fingerprinting, selección del módulo, comprobación de requisitos, prueba controlada y cierre de cualquier sesión generada.

Qué hace diferente este ejemplo

Diferencia técnica: afecta a infraestructura de balanceo/entrega, por lo que una validación profesional debe coordinarse para evitar impacto sobre tráfico real.

Defensa y cierre

Prioridad: aplicar mitigaciones/actualizaciones del proveedor y revisar el dispositivo antes de devolverlo a producción.

SECUENCIA DE CONSOLA - PLANTILLA SEGURA

```
search cve:<CVE>
info <modulo>
use <modulo>
show options
set RHOSTS <IP_VM_LAB>
check   # si está soportado
back
```

CASO 5 - Google Chromium V8

CVE-2026-11645

Qué ocurrió

Lectura y escritura fuera de límites en V8. CISA confirmó explotación activa; NVD la describe como una vulnerabilidad de memoria en Chromium.

Flujo de Metasploit en laboratorio

Familia Metasploit: conceptualmente corresponde a explotación de cliente/navegador, distinta de los cuatro casos de servidor. En laboratorio requiere una versión vulnerable aislada y una página/servidor de prueba controlados.

Qué hace diferente este ejemplo

Diferencia técnica: corrupción de memoria en un cliente. Arquitectura, versión exacta y mitigaciones del navegador son determinantes; no debe trasladarse una PoC a usuarios reales.

Defensa y cierre

Prioridad: actualizar Chromium/Chrome a una versión corregida y retirar versiones vulnerables.

SECUENCIA DE CONSOLA - PLANTILLA SEGURA

```
search cve:<CVE>
info <modulo>
use <modulo>
show options
set RHOSTS <IP_VM_LAB>
check   # si está soportado
back
```

NOTAS Y FUENTES DE LOS CINCO CASOS

• CISA KEV, junio-agosto de 2026: confirma explotación activa de CVE-2026-58644, CVE-2026-25089, CVE-2026-15410, CVE-2026-8037 y CVE-2026-11645.

• NVD: CVE-2026-58644 figura con explotación activa, automatizable e impacto técnico total; CVE-2026-11645 corresponde a lectura/escritura fuera de límites en Chromium V8.

• Los nombres exactos de módulos de Metasploit pueden cambiar con la versión del Framework. La guía enseña a localizarlos con 'search' y a verificar 'info' en la instalación actual, en vez de fijar rutas que puedan quedar obsoletas.

• Esta sección diferencia cinco superficies/técnicas: deserialización web, command injection en sandbox, code injection en appliance VPN, command injection en infraestructura de balanceo y corrupción de memoria en navegador.

CRITERIO EDITORIAL

Se han escogido vulnerabilidades recientes con evidencia pública de explotación para que el lector comprenda cómo cambia la metodología según la tecnología afectada, manteniendo el contenido en un nivel de laboratorio responsable.

AMPLIACION TECNICA I

Arquitectura de Metasploit Framework y modelo mental de trabajo

Metasploit Framework es una plataforma modular de pruebas de penetracion. Su potencia no reside en un unico exploit, sino en la forma en que organiza reconocimiento, validacion, explotacion controlada y acciones posteriores dentro de una interfaz coherente. El objetivo profesional es transformar datos tecnicos en una prueba reproducible y documentada, siempre dentro del alcance autorizado.

1. Componentes principales

• **Exploit:** modulo que implementa la logica necesaria para comprobar o aprovechar una vulnerabilidad concreta.

• **Auxiliary:** tareas de apoyo como descubrimiento, enumeracion y comprobaciones que no necesitan entregar un payload.

• **Payload:** codigo o accion que se ejecuta cuando un exploit obtiene la capacidad prevista. La compatibilidad depende del modulo y del objetivo.

• **Post:** modulos diseñados para trabajar sobre una sesion ya establecida y obtener informacion util para la auditoria.

• **Encoder y NOP:** componentes especializados de la arquitectura; no deben confundirse con mecanismos universales para eludir controles.

2. El datastore: configuracion local y global

Las opciones de un modulo se almacenan en el datastore. **set** modifica una opcion para el modulo actual; **setg** establece un valor global reutilizable. En una auditoria conviene minimizar los valores globales para evitar que una configuracion de un ejercicio termine aplicandose accidentalmente a otro.

Comandos de orientacion seguros

```
help
search type:auxiliary
search cve:AAAA-NNNN
info <modulo>
show options
show payloads
back
```

3. Lectura profesional de un modulo

Antes de ejecutar nada, revise nombre, descripcion, plataforma, referencias, opciones requeridas, objetivos compatibles y comportamiento esperado. La existencia de un modulo no demuestra por si sola que el sistema sea vulnerable. Version, arquitectura, configuracion, mitigaciones y parches pueden cambiar completamente el resultado.

REGLA DE LABORATORIO: mantenga atacante y objetivo en una red virtual aislada/host-only, utilice snapshots y no conecte una maquina deliberadamente vulnerable directamente a Internet.

Fuente principal: documentacion oficial de Rapid7 Metasploit Framework (arquitectura, modulos y msfconsole).

AMPLIACION TECNICA II

msfconsole: busqueda, seleccion y validacion

La consola es la interfaz central del Framework. Una metodologia madura evita comenzar por 'run'. Primero se formula una hipotesis basada en evidencias: servicio detectado, version probable, vulnerabilidad asociada y modulo candidato.

Flujo recomendado

• Identifique el activo dentro del alcance y documente su direccion, sistema operativo esperado y finalidad del laboratorio.

• Enumere servicios de forma proporcionada. Registre puerto, protocolo, producto y version cuando sea posible.

• Busque modulos mediante filtros: nombre, plataforma, tipo o identificador CVE.

• Abra la informacion del modulo y contraste sus referencias con la version realmente observada.

• Revise **show options**. No ejecute con parametros heredados sin comprobarlos.

• Cuando el modulo lo soporte, utilice una comprobacion no destructiva antes de cualquier intento de explotacion.

• Registre hora, modulo, parametros esenciales, resultado y evidencia.

Ejemplo de investigacion sin explotacion

```
search type:auxiliary name:smb
search type:exploit platform:linux
info <ruta_del_modulo>
use <ruta_del_modulo>
show options
show advanced
```

Interpretacion de resultados

Un resultado negativo tampoco siempre significa 'seguro'. Puede existir filtrado, deteccion incorrecta de version, dependencia de una configuracion concreta o un falso negativo. Del mismo modo, un banner antiguo no basta para afirmar vulnerabilidad. La conclusion debe combinar evidencia de red, informacion del modulo y validacion independiente.

Documentacion minima por prueba

• Activo y propietario autorizado; fecha y ventana de prueba.

• Servicio/version observados y metodo usado para identificarlos.

• Modulo candidato y razon tecnica de seleccion.

• Opciones modificadas respecto a los valores por defecto.

• Resultado: no aplicable, no vulnerable, vulnerable validado o inconcluso.

• Capturas/logs estrictamente necesarios y recomendacion de remediacion.

La calidad de una auditoria depende tanto de poder reproducir el hallazgo como de limitar el impacto de la prueba.

AMPLIACION TECNICA III

Payloads, sesiones y control del impacto

Un exploit y un payload son conceptos distintos. El exploit intenta alcanzar una condicion vulnerable; el payload define la accion posterior compatible. Metasploit puede mostrar los payloads compatibles con un modulo, lo que evita seleccionar combinaciones que no corresponden a la plataforma o arquitectura.

Staged y stageless

En terminos conceptuales, un payload **staged** separa la entrega en fases; uno **stageless** contiene la funcionalidad en una unica carga. La eleccion afecta tamaño, fiabilidad, conectividad y trazabilidad. En un laboratorio educativo interesa comprender esas diferencias, no intentar ocultar actividad.

Sesiones

Tras una validacion exitosa puede aparecer una shell convencional o una sesion Meterpreter. La documentacion de Rapid7 distingue ambas: una shell ofrece la interfaz de comandos propia del objetivo; Meterpreter integra funciones adicionales de Metasploit. En un ejercicio autorizado, la sesion debe considerarse evidencia temporal y manejarse con el minimo privilegio necesario.

Comandos de gestion no destructiva

```
sessions
sessions -l
background
sessions -i <ID>
exit
```

Principios de post-explotacion responsable

• Recolecte solo la informacion necesaria para demostrar impacto.

• No copie datos personales o secretos reales cuando una evidencia sintetica sea suficiente.

• No instale persistencia salvo que forme parte expresa del alcance y exista un procedimiento de restauracion.

• No altere cuentas, configuraciones, registros o servicios productivos para 'demostrar' control.

• Mantenga un registro de cambios y cierre las sesiones al finalizar.

• Restaure snapshots del laboratorio y compruebe que no quedan listeners, procesos o artefactos.

Bind frente a reverse

Ambos describen modelos de conectividad. En un laboratorio aislado sirven para comprender como las reglas de red y NAT afectan al establecimiento de una sesion. La configuracion concreta debe diseñarse para que el trafico nunca abandone el segmento de pruebas.

Una demostracion profesional prueba el riesgo con el menor impacto posible. Obtener mas acceso del necesario no mejora el hallazgo.

AMPLIACION TECNICA IV

Metodologia de pentesting con Metasploit

Metasploit funciona mejor cuando se integra en una metodologia, no cuando se utiliza como una coleccion de exploits. El ciclo siguiente convierte una prueba tecnica en una auditoria defendible.

FASE	OBJETIVO	EVIDENCIA
1. Alcance	Definir activos, limites y autorizacion	Documento de alcance
2. Descubrimiento	Identificar hosts y servicios	Inventario tecnico
3. Enumeracion	Precisar versiones/configuracion	Banners y resultados
4. Correlacion	Relacionar evidencia con vulnerabilidades	CVE/advisory/modulo
5. Validacion	Confirmar de forma controlada	Salida reproducible
6. Impacto	Demostrar consecuencia minima necesaria	Evidencia limitada
7. Limpieza	Cerrar sesiones/restaurar entorno	Checklist de cierre
8. Informe	Explicar riesgo y solucion	Hallazgo y remediacion

Modelo de hallazgo

```
Titulo: servicio vulnerable o configuracion insegura.
Activo: identificador del sistema de laboratorio.
Evidencia: version, respuesta o sesion obtenida.
Impacto: consecuencia tecnica demostrada sin extrapolaciones innecesarias.
Severidad: basada en contexto, explotabilidad y consecuencias.
Remediacion: parche, actualizacion, deshabilitacion del servicio, segmentacion o endurecimiento.
Retest: repetir la comprobacion despues de aplicar la correccion.
```

Automatizacion mediante resource scripts

Metasploit admite scripts de recursos que ejecutan secuencias de comandos. Son utiles para repetir configuraciones de laboratorio, pero tambien multiplican los errores: un parametro incorrecto puede repetirse automaticamente. Versione los scripts, documente cada comando y mantenga listas de objetivos separadas.

No automatice una accion destructiva simplemente porque la herramienta lo permite. La automatizacion debe reducir errores y aumentar reproducibilidad.

AMPLIACION TECNICA V

Laboratorio controlado con Metasploitable: practica guiada segura

Metasploitable es una maquina deliberadamente vulnerable destinada al aprendizaje. El siguiente ejercicio se centra en el proceso profesional: descubrir, investigar y preparar una validacion sin proporcionar una cadena de intrusiones reutilizable contra sistemas ajenos.

Topologia sugerida

```
Kali/Metasploit 192.168.56.10 <-- red HOST-ONLY --> 192.168.56.20 Metasploitable
Sin bridge. Sin port-forwarding. Snapshot de ambas maquinas antes de comenzar.
```

Paso 1 - Verifique el aislamiento

Compruebe en el hipervisor que ambas interfaces pertenecen exclusivamente a la red de laboratorio. El objetivo deliberadamente vulnerable no debe tener una interfaz puenteada hacia la LAN domestica/empresarial ni exposicion directa a Internet.

Paso 2 - Construya el inventario

Desde el sistema auditor identifique que el objetivo responde y enumere exclusivamente la IP de laboratorio. Anote los servicios encontrados y sus versiones aparentes. El resultado sera su tabla de hipotesis, no una autorizacion para ejecutar todos los exploits disponibles.

Paso 3 - Correlacione un servicio con Metasploit

```
msfconsole
search name:<servicio_observado>
info <modulo_candidato>
show options
```

Paso 4 - Compare requisitos

Lea la descripcion y referencias. Compare plataforma, version, puerto esperado y condiciones previas con lo observado. Si no coinciden, descarte el modulo. Este paso evita la practica habitual -y incorrecta- de probar exploits al azar.

Paso 5 - Validacion de bajo impacto

Si el modulo dispone de una funcion de comprobacion segura, utilicela en el laboratorio y documente el resultado. Si la validacion exige explotacion, defina previamente que evidencia minima necesita obtener y detengase al conseguirla.

Paso 6 - Cierre y retest

Cierre cualquier sesion creada, restaure el snapshot del objetivo y documente la mitigacion que corresponderia en un sistema real: actualizacion, retirada de software obsoleto, restriccion del puerto, segmentacion o sustitucion del servicio.

Checklist final

- ¿Existe autorizacion y alcance escrito?
- ¿El objetivo esta completamente aislado?
- ¿La version observada coincide con los requisitos del modulo?
- ¿Se ha elegido la comprobacion menos invasiva?
- ¿La evidencia obtenida es suficiente sin acceder a datos innecesarios?
- ¿Se han cerrado sesiones y restaurado snapshots?
- ¿El informe explica una remediacion verificable?

Referencias de esta ampliacion: documentacion oficial de Rapid7 sobre Framework, Modules, Manual Exploitation, Payloads, Sessions y Post-Exploitation.

EJEMPLOS DE VULNERABILIDADES DETECTADAS EN MÁQUINA VIRTUAL METASPLOITABLE

Análisis defensivo, detección y mitigación

Esta sección reorganiza el material aportado sobre vulnerabilidades presentes en entornos tipo Metasploitable y lo integra visualmente con las páginas de ejemplos de explotación de la guía. El enfoque se mantiene en identificación, validación controlada y bastionado.

ENTORNO DE PRÁCTICA

Utilizar únicamente una máquina virtual propia o expresamente autorizada, preferiblemente en una red host-only o aislada. Las versiones deliberadamente vulnerables de Metasploitable no deben exponerse directamente a Internet.

Contenido de esta sección

1. FTP - vsftpd 2.3.4 Backdoor
2. SMB - Samba Username Map Script (CVE-2007-2447)
3. IRC - UnrealIRCd 3.2.8.1 Backdoor
4. distcc (CVE-2004-2687)
5. NFS - exportación insegura del directorio raíz

La estructura de cada caso es uniforme: análisis del fallo, detección en laboratorio, interpretación de la evidencia y mitigación.

CASO 1 - Servicio FTP: vsftpd 2.3.4 Backdoor

Análisis del fallo

El incidente corresponde a una alteración de cadena de suministro. La versión comprometida contenía un backdoor: un nombre de usuario que incluyera la secuencia `:)` podía provocar la apertura silenciosa del puerto 6200 y proporcionar una shell con privilegios elevados.

Detección en la máquina virtual

```
analista@soc:~$ nmap -sV -p 21 192.168.1.100
Starting Nmap 7.93...
21/tcp open ftp vsftpd 2.3.4

analista@soc:~$ nmap --script ftp-vsftpd-backdoor -p 21 192.168.1.100
```

Qué debe registrar el auditor

Servicio y versión observados, puerto, método de detección, resultado, fecha de la prueba y evidencia mínima necesaria para poder repetir el hallazgo.

Mitigación y bastionado

- Detener inmediatamente el servicio vulnerable: `systemctl stop vsftpd`.
- Actualizar el paquete a una versión segura y soportada.
- Migrar a SFTP sobre SSH cuando sea posible, evitando FTP tradicional en texto plano.

Criterio de cierre

Aplicar la corrección, repetir la comprobación y conservar evidencia de que el servicio vulnerable o la configuración insegura ya no está presente.

Nota: ejemplo destinado a laboratorio Metasploitable y análisis defensivo autorizado.

CASO 2 - Servicio SMB: Samba Username Map Script

Análisis del fallo

CVE-2007-2447. El problema se relaciona con la opción `username map script` de Samba. En configuraciones vulnerables, datos introducidos en el campo de usuario podían alcanzar la ejecución de comandos antes de la validación de contraseña.

Detección en la máquina virtual

```
analista@soc:~$ nmap --script smb-os-discovery -p 139,445 192.168.1.100
...
OS: Unix (Samba 3.0.20-Debian)
```

Qué debe registrar el auditor

Servicio y versión observados, puerto, método de detección, resultado, fecha de la prueba y evidencia mínima necesaria para poder repetir el hallazgo.

Mitigación y bastionado

- Revisar `/etc/samba/smb.conf`.
- Localizar y deshabilitar `username map script` si no es imprescindible.
- Reiniciar el servicio tras la corrección y actualizar Samba a una versión soportada.

Criterio de cierre

Aplicar la corrección, repetir la comprobación y conservar evidencia de que el servicio vulnerable o la configuración insegura ya no está presente.

Nota: ejemplo destinado a laboratorio Metasploitable y análisis defensivo autorizado.

CASO 3 - Servicio IRC: UnrealIRCd 3.2.8.1 Backdoor

Análisis del fallo

El material identifica otro incidente de cadena de suministro. La distribución comprometida de UnrealIRCd 3.2.8.1 permitía que determinadas entradas recibidas por el servicio terminaran ejecutándose con los privilegios del demonio.

Detección en la máquina virtual

```
analista@soc:~$ nmap -sV -p 6667 192.168.1.100
6667/tcp open irc UnrealIRCd 3.2.8.1
```

Qué debe registrar el auditor

Servicio y versión observados, puerto, método de detección, resultado, fecha de la prueba y evidencia mínima necesaria para poder repetir el hallazgo.

Mitigación y bastionado

- Detener el servicio IRC vulnerable.
- Instalar una versión limpia obtenida desde una fuente de confianza.
- Verificar hashes o firmas criptográficas publicadas oficialmente antes de desplegar software.

Criterio de cierre

Aplicar la corrección, repetir la comprobación y conservar evidencia de que el servicio vulnerable o la configuración insegura ya no está presente.

Nota: ejemplo destinado a laboratorio Metasploitable y análisis defensivo autorizado.

CASO 4 - Compilación distribuida: distcc

Análisis del fallo

CVE-2004-2687. Versiones antiguas de distcc confiaban ampliamente en la red y podían operar sin autenticación. Cuando el servicio queda accesible desde una red no confiable, esa arquitectura puede permitir el envío de trabajos que incorporen comandos arbitrarios.

Detección en la máquina virtual

```
analista@soc:~$ nmap --script distcc-cve2004-2687 -p 3632 192.168.1.100
```

Qué debe registrar el auditor

Servicio y versión observados, puerto, método de detección, resultado, fecha de la prueba y evidencia mínima necesaria para poder repetir el hallazgo.

Mitigación y bastionado

- No exponer distcc a Internet.
- Restringir mediante firewall las IP que realmente necesitan acceder al servicio.
- Cuando proceda, transportar el tráfico mediante canales cifrados y autenticados como SSH.

Criterio de cierre

Aplicar la corrección, repetir la comprobación y conservar evidencia de que el servicio vulnerable o la configuración insegura ya no está presente.

Nota: ejemplo destinado a laboratorio Metasploitable y análisis defensivo autorizado.

CASO 5 - Configuración insegura: NFS exportando '/'

Análisis del fallo

Este caso es una configuración insegura grave. El directorio raíz `/` aparece exportado y el escenario descrito desactiva `root_squash`, lo que puede permitir que un cliente con privilegios elevados modifique archivos críticos del sistema.

Detección en la máquina virtual

```
analista@soc:~$ showmount -e 192.168.1.100
Export list for 192.168.1.100:
/ *
```

Qué debe registrar el auditor

Servicio y versión observados, puerto, método de detección, resultado, fecha de la prueba y evidencia mínima necesaria para poder repetir el hallazgo.

Mitigación y bastionado

- Editar `/etc/exports` y exportar solo los directorios estrictamente necesarios.
- Restringir el acceso a rangos de red de confianza en lugar de utilizar un asterisco.
- Mantener `root_squash` activado.
- Aplicar la configuración corregida con `exportfs -ra`.

Criterio de cierre

Aplicar la corrección, repetir la comprobación y conservar evidencia de que el servicio vulnerable o la configuración insegura ya no está presente.

Nota: ejemplo destinado a laboratorio Metasploitable y análisis defensivo autorizado.

AGRADECIMIENTOS

Una guía técnica nunca nace únicamente de comandos, documentación o pruebas de laboratorio. También nace de la curiosidad, de las horas dedicadas a comprender por qué algo funciona y, sobre todo, de la voluntad de compartir lo aprendido.

Mi agradecimiento a la comunidad de ciberseguridad y software libre, a investigadores, desarrolladores, administradores de sistemas, docentes y profesionales que publican conocimiento técnico y hacen posible que nuevas generaciones puedan aprender sobre seguridad de una forma cada vez más rigurosa.

También a quienes trabajan cada día en defensa, respuesta a incidentes y hacking ético. Su trabajo recuerda que detrás de cada vulnerabilidad existen sistemas, organizaciones y personas que dependen de que la tecnología sea más segura.

Bussio28Team nace precisamente de esa idea: seguridad, conocimiento y libertad responsable. Esta guía pretende aportar una pequeña pieza a ese objetivo, acercando Metasploit desde una perspectiva práctica, profesional y ética.

Y, especialmente, gracias a ti, lector. Si estas páginas te llevan a investigar un poco más, a montar tu propio laboratorio, a documentar mejor una auditoría o a comprender una vulnerabilidad que antes parecía inaccesible, el trabajo habrá merecido la pena.

«El conocimiento se multiplica cuando se comparte; la seguridad mejora cuando ese conocimiento se utiliza con responsabilidad.»

Antoni Clarens
Bussio28Team