FORTRA™

Outflank OST Modules Technical Overview

June 2024

Outflank Security Tooling (OST) is a set of private offensive security tools created by the red teaming specialists of Outflank, covering all significant steps in the attacker kill chain, scrutinized, and made available for use by vetted red teams. While this document focuses on the various tools available in Outflank OST, another valuable component of the solution is the shared background tradecraft knowledge as well as the ever-growing Outflank community in the Outflank Slack community. The Outflank OST Portal includes additional resources such as video recordings of the OST toolset, quarterly video updates, and Tech Deep Dive sessions on various related topics.

The Outflank team regularly presents at various industry events and has provided trainings to the community – most recently an MS Office Tradecraft session with over 1,000 registered attendees.

The Outflank tools allow you to simulate similar techniques to what some APTs and Organized Crime Groups apply but are not available in public tools. They also help your team members to easily perform deep technical and difficult tasks without hassle, while being OPSEC safe. OST tools are explicitly developed to help enable a skilled operator bypass defensive measures and detection tools.

The tools are categorized along the phases of a typical attack kill chain:

- In Phase (initial access)
- Through Phase (C2, lateral movement, privilege escalation)
- Out Phase (actions on objectives)

In addition, other categories of tools are provided:

- EDR Evasion
- Support (scripts and tools to support the red teaming process)
- Miscellaneous (various smaller attack tools and scripts)
- Cobalt Strike Enhancer: Beacon Booster

In the remainder of this document, a high-level technical overview of the tools in OST is provided.

EDR Evasion

Background and Concepts

Endpoint Detection and Response (EDR) products typically consist of a centralized server and a collection of endpoint agents. Agents comprise multiple sensory components, including a DLL loaded into each process, a kernel driver, or a startup service for uploading telemetry and directing the agent. Each agent collects information from the following telemetry sources and acts based on local logic or as instructed by the server.

Techniques that bypass all EDRs in one go are now increasingly less common. It is becoming more important for red teamers to know which exact technique to use (or not use) for a specific EDR. Up-to-date knowledge is important—particularly knowledge on how a specific EDR gets its data, how it works under the hood and where its blind spots and bypass opportunities lie. For OST we aimed on better equipping our customers with exactly that: knowledge. We started with what we call “EDR Presets” in our Payload Generator. Payload Generator allows red team operators to easily create payloads with a wide range of decoy tricks, binary transformation options (such as prepending random shellcode), payload transformation options (such as encoding), encryption and compression, encryption possibilities, process creation techniques (such as Write Hook, KernelCallbackTable, Earlybird, in-process thread), as well as OPSEC tricks, guardrails and many more. The total set of option combinations easily surpasses a hundred, and it is critical to select the proper combination to bypass specific EDRs. EDR Presets are configuration sets of these 100+ settings in the Payload Generator. Each EDR Preset has demonstrated the ability to evade detection by a certain EDR system at specific points in time. Though presets are incredibly useful and are a significant time saver for red team operators, they do have a limited shelf life. In order to maintain an up-to-date library of effective presets, we obviously perform ongoing lab testing. But as OST has a large community of active users, we decided to use that power of the community. We have implemented a secure way where OST users can contribute such presets they discovered during their engagements. This collaborative effort highlights the benefit of being a part of a strong

user community, providing access to a collective pool of knowledge from which everyone can benefit.

In Phase

PE Payload Generator

Payload generator is a binary payload builder/transformer focused on OPSEC safety, operational traceability, and anti-forensics. It can generate various output formats useful for many red teaming use cases, including:

- Phishing
- Dropper
- Persistency
- Privilege Escalation
- Lateral movement
- Legup provider
- UAC bypass
- AppLocker bypass

The features provided should work on arbitrary payloads but are tested on Cobalt Strike and Stage1 payloads. The build process is always performed using OPSEC safe parameters (release, no debug, etc.) and all outputs, together with an IOC overview are provided as output.

The general workflow for an operator is to:

- Select the right project
- Upload a payload (raw, x86/x64) generated by Cobalt Strike, Stage1 or a custom payload, or alternatively point to a staging location.
- Configure output formats and transformations

Payload generator offers the latest offensive R&D to help bypassing AV and EDR products, such as direct system calls, techniques to blend in with TI ETW, ROP gadgets, sleep masking, stack spoofing and much more.

Once the configuration is complete the operator can start a build. This will generate the final payload in the format requested and provide various metadata for payload management (e.g. tracking exact options configured in this specific payload).

In-Phase Builder (BETA)

The In-phase builder can be used for transforming a binary payload in various output formats including script based formats (e.g. HTA, VBScript and JScript) as well as more complex file formats and payload-wrapper formats (e.g. CHM, ClickOnce). Intent of builder is to make sophisticated, multilayered phishing samples with a couple of clicks.

Using Builder, an operator can convert into various other formats. Shellcode can be converted into various formats, and each format can allow for new conversions. For example:

- Shellcode can be converted into ClickOnce, this is deemed a 'final format' and does not allow further conversions. ClickOnce is a deployment technology that allows users to run (and optionally install) applications from a web server.
- Shellcode can be converted into JScript or VBScript, which allows for new conversions (e.g. CHM, HTA, SCT, WSF, MSC).

This tool is under active development and will be evolving over time with additional features and file-format releases.

Stego Loader

Stego loader allows the operator to embed a payload into an image file using Least Significant Bit Steganography. After the payload is embedded into the image, the image is still valid and viewable. Host the image with the hidden payload on a redirector and have a stager retrieve it. The result will be decoded (and optionally decrypted) before execution/dropping/loading. Encoding is just a

means to obfuscate to bypass automated detections. This does not provide any safety, only obscurity.

Office Intrusion Pack

The Office Intrusion Pack generates VBA macros based on various templates and settings. Currently it generates a .txt file with VBA code. The main use case is generating malicious macros to be used in maldocs for initial access or persistence (e.g. by backdooring normal.dotm). The templates used for generating the macro payloads are based on various techniques observed in the wild, including techniques used by APTs (such as UUID encoding of payloads and callback thread execution). Some of these include:

- Bypassing Mark of the web restrictions in Office - Certain customers have hardened Office and enabled a policy which blocks macro execution for downloaded files. We have strategies to bypass these restrictions.
- Bypass macro signing - approaches that can be taken when you encounter a target that solely accepts signed macros.
- Lateral movement via Office macros - Various companies struggle with securing Office macros. Specifically in the financial industry, companies are quite risk averse in hardening macros as they are often widely used in their business. This writeup shows various attack patterns to exploit weak Office settings.
- Macroless Office reconnaissance - Sometimes you want to obtain information on a workstation without running an Office macro. Specifically, you may need an exact Office version so that an EvilClippied macro can be delivered later.

Language Panda

Language Panda changes language-artifacts within your office document to make it appear as if it was created using a Russian / Hebrew / Ukrainian / etc. MS Office installation.

- Change the language characteristics of a Word, Excel, or PowerPoint document.

- Extendable with various languages: Russian released in first version. Later Hebrew, Ukrainian, etc..
- Support for all Office Open XML (OpenXML or OOXML) Word, Excel and PowerPoint filetypes: docx, docm, dotx|m, xlsx|m, xlam, xlsb, pptx|m, etc.

Planting these false flags in your MS Office payload may help making your red teaming operation to appear to be more realistic to a responding blue team.

Through Phase

Stage 1

Stage1 is Outflank's custom Command-and-Control framework with focus on OPSEC and 'Stage 1' functionality. Stage1 is a light C2 framework that is aiming to be as OPSEC-safe as possible. Stage1 uses features such as direct system calls and sleep masking to stay under the radar of AV and EDR for your initial access and local reconnaissance activities. In addition, it offers various advanced process injection techniques and thread management features to help stealthily transition to a Stage 2 C2 such as Cobalt Strike. In our latest release, the speed of SOCKS tunneling has been greatly accelerated.

Lateral Pack - ShovelNG

ShovelNG is an our most OPSEC safe toolkit for lateral movement in Windows networks. It abuses different techniques for remote code execution such as WMI, WSMan and DCOM. It also allows flexibility to the red team operator to select relevant LOLBAS and select the loader to be usedIt integrates with Cobalt Strike and Stage1 but can be used with any Command and Control framework with BOF support.
ShovelNG's functionality can be extended for bypassing UAC or perform Local Privilege Escalation, by using the added SCMUseKerberos tool.

Lateral Pack – pyShovel

pyShovel is a a python implementation of some of the ShovelNG functionality. It has added use in cases where target networks are tightly firewalled and only WMI/RPC and SMB protocol functionality is available. It leverages the Impacket libraries to do so. In order to ensure proper functioning this tool is embedded in a docker environment that will install all the needed libraries.

SharpFuscator

SharpFuscator is a .NET obfuscator which compiles C# repositories, source code or binaries making the following change to source code:

- Names of namespaces, classes, delegates and interfaces will be replaced with randomly generated names
- Strings will be encrypted and a decryption function will be added to main class
- For some projects, code will be patched. I.e. Rubeus is rewritten so that all Rubeus LDAP queries communication is encrypted.
- Array initializations can be reversed (optional)
- Check for debugger presence can be added (optional)
- Check for sandbox artifacts presence can be added (optional)

For some tools we have noticed that even the XORed version of specific strings is flagged by AV/EDR. In these cases it is possible to use AES for strings encryption.

If the use of Costura is detected, it will be reimplemented in the output assembly. For this task Mono.Cecil | Mono is used to inject a module initializer to be able to resolve references from embedded resources Module initializers in C#.

ShapFuscator is designed for AV/EDR static signature evasion. By analyzing the assembly with tools such as dnspy it is relatively easy to identify the encryption keys, the assembly features or the original tool.

Credential Pack

Credential Pack is a collection of tools that can be used with any Command and Control framework with BOF support and allows the red team operators to extract (dump) and obtain credentials.

The toolset currently consists of the following tools:

- **DumpertNG**: an evolved and improved version of the original public Dumpert tool from outflank. DumpertNG uses a `PROCESS_CREATE_PROCESS` handle to create a process snapshot and uses direct system calls and API unhooking to dump LSASS memory from the snapshot handle.
- **ProcessDupMiniDump**: LSASS dump tool that uses a `PROCESS_CREATE_PROCESS` handle to create a forked LSASS process and uses direct system calls and API unhooking to dump LSASS memory from the forked process handle.
- **HandleDupMiniDump**: LSASS dump tool that enumerates active processes on a system for existing LSASS handles and "when found" duplicates an existing handle and use direct system calls and API unhooking to dump LSASS memory from the existing process handle.
- **DecompreZz**: the produced LSASS memory dump files from the "userland" LSASS dump tools above are compressed using native API calls and saved to a random filename within the Windows temp folder. An additional decompress tool is provided to decompress and obtain the original memory dump file.
- **KernelKatz**: LSASS dump tool which uses a technique called "Bring Your Own Vulnerable Driver" (BYOVD) to install a "vulnerable" signed kernel driver on the system, which we can abuse to dump credentials from the LSASS process memory within kernelspace.
- **PasswordSpy++**: Our implementation of NPLogonNotify with AES encryption and SACLS on the output file. Every password is encrypted using a different key written to a file

in `C:\Windows\Temp\` with a read|del everyone, write admin SACL. A python script allows decryption.

DLL Hijack Library – Local Privilege Easy (LPEasy)

Local Privilege Easy (LPEasy) is a DLL hijacking project to facilitate Local Privilege Escalation (LPE) attacks.

This project helps you in abusing DLL hijacks on a system. It solely helps in the abuse, not the detection process of a DLL hijack.

Out Phase

Hidden Desktop

Hidden Desktop provides a full graphical desktop experience to a remote system, without the victim knowing. Ever required access to a fat-client application that is running on the victim's system in an OUT phase? Using Hidden Desktop you can spawn a new desktop on a user's system and redirect it to you. You can use it to open the Explorer, or for example any browser installed on the system.

- Hidden Desktop is a feature in top-tier financial malware
- Usually referred to as HVNC or HiddenVNC on darknet markets
- Despite the name Hidden Desktop is NOT relying on the VNC protocol but uses specific Windows APIs that are supported since Windows 2k.
- Malware family examples using similar functionality: Dridex, Gozi

Fake Ransom

FakeRansom is a tool developed and used to support in the out phase of a ransomware attack simulation. It is fake, yet real-life-like ransomware. It hijacks the screen and shows a full screen ransom notice combined with ongoing file listings of files of that computer. It also blocks common

keyboard combinations (i.e. `ALT-TAB` and `CTRL-ALT-DEL`) of a user trying to exit the ransom notice. It creates the sense of urgency and a stress factor that is often overlooked in a simulation.

Support Phase

BlueCheck

BlueCheck is a collection of tools that help you identify possible Blue team activity. It integrates with RedELK. It is compatible with Cobalstrike using reflective DLLs or by using Beacon Object Files (BOF). The reflective DLLs can also be used in other C2 frameworks, which support reflective DLL injection.

The toolset currently consists of the following tools:

- **CertCheck**: enables the RedTeam operator to check SSL certificate information (SSL interception e.g.) from a specified website.
- **PasswordChangeCheck**: enables the RedTeam operator to check specific Active Directory accounts for password changes.
- **SecurityToolCheck**: shows information about security tools and products running on the system.

BeaconBot

BeaconBot is a Cobalt Strike CNA script that performs all kind of administrative duties. You can consider this bot to be your Cobalt Strike secretary.

The most important functions include:

- Administration of beacons
    - Beacon notifications and digest
    - Track sleep and late beacons
- Triage of incoming beacons

- Auto sleep beacons
- Auto exit beacons
- Auto shinject beacons

The triage commands are specifically of value when a 'user behaviour based persistence' (e.g. application startup) is applied and as such 'beacon storms' can occur when a user starts an application many times.

Cobalt Strike Enhancement

User Defined Reflective Loader

A Reflective Loader turns a compiled implant (DLL) into Position Independent Shellcode (PIC). When using Cobalt Strike without a custom reflective loader, the Cobalt Strike default reflective loader is used. This loader is signatured and could result in unwanted detections. Within Cobalt Strike, it is possible to configure a custom User Defined Reflective Loader (UDRL).
Using a custom UDRL can avoid detection. The native option to configure a UDRL is to rely on the Cobalt Strike Arsenal kit and Visual Studio. Within his UDRL module for Cobalt Strike, we have taken a slightly different approach.
Within OST an operator can provide a Cobalt Strike raw/bin files and OST will patch an UDRL into it. Furthermore, we run some YARA rules and try to rewrite the implant's code to bypass YARA rules. The resulting shellcode can be used in further OST processes (e.g. PE Payload Generator).

Cobalt Strike Enhancer: Beacon Booster

Fortra's Cobalt Strike is a very customizable adversary simulation tool. To improve evasion, it is recommended to customize the beacon implant to your situation. The beacon can be customized using the Aggressor script as well as the various kits in Arsenal kit. From the Arsenal Kit, two elements stand out in importance for OPSEC safety; the Reflective Loader (UDRL) and the Sleep Mask.

Using a custom UDRL and Sleep Mask can avoid detection. The native option to configure these is to rely on the Cobalt Strike Arsenal kit and develop your own in C / C++ / Visual Studio. Within his UDRL module for Cobalt Strike, we have take a slightly different approach. Within OST an operator can provide a Cobalt Strike raw/bin files and OST will patch an UDRL into it.

The main benefit of Beacon Booster is to use more advanced evasion features of Cobalt Strike without having to write custom code or touching Visual Studio. The standard loader of Cobalt Strike may be signatured by AV, where custom loaders are likely not.

Also, before the UDRL and Sleep Mask are created, various known (YARA) IOC's are replaced with custom variants. This allows the resulting bin to bypass more signatures.

Miscellaneous

UntrustProcess

A BOF that downgrades a process token to Untrusted integrity level. Requires high integrity token (admin privileges). The principal you are executing as must have control over the target process token.

KerberosAsk

Implementation of various Rubeus-alike Kerberos functionality in BOF format. It uses a custom implementation for ASN1 encoding. This BOF focusses on ticket requests and renewals. It allows you to perform offensive Kerberos operations without having to do process injections and run Rubeus' .NET code.

Coercer

Collection of various methods to coerce a system to perform authentication towards another system (generally, a relaying server).

In many cases you will want your relaying server to relay the incoming authentication packets to an ADCS HTTP endpoint (in case of SMB connections) or LDAP for RBCD/ShadowCred abuse.

O365TokenExtractor

A BOF that dumps available O365 tokens stored for the current user by the Web Account Manager.

The WAM (Web Account Manager) component stores O365 access tokens in a cache that is accessible for the current user. Running applications that interact with Office365 (e.g. MS Teams) will store their tokens in this cache.

MSAL (Microsoft Authentication Library) is able to call WAM, a Windows 10 component that ships with the OS. This component acts as an authentication broker and allows applications to piggyback on the authenticated Windows session.

O365TokenExtractor can dump the available tokens for the current user.

KernelTool

KernelTool is toolset that uses a kernel driver helper to interact with the system. The toolset includes commands for setting and removing process protections and modifying callbacks. It uses read/write primitives in the vulnerable the kernel driver to modify kernel memory to perform these actions. It comes with a vulnerable driver that is not on the Microsoft block list.

SideloadTrigger

SideloadTrigger is a collection of methods to trigger sideloads. E.g. to trigger a DLL sideload in a Windows service through a hijackable PATH.

RPC and Registry Tradecraft

Collection of miscellaneous scripts that we create during research time, ad-hoc during projects, or for talks/presentations. This collection may be extended at any time.

Dump Mstsc

Dump credentials of RDP client processes on the system. Can decrypt credentials of the current user's context when in medium integrity, and credentials of other users when in high integrity.

KernelKatz

KernelKatz allows reading LSASS memory through the kernel by leveraging a kernel driver and can dump hashes of login sessions. It comes with a vulnerable driver that is not on the Microsoft block list.

EvilClicky

EvilClicky is a Python-based ClickOnce payload generator that creates the necessary ClickOnce files for initial execution. It leverages a trusted executable to bypass SmartScreen controls.

Outflank – C2 Tool Collection

On GitHub, we maintain a public repository of Red Teaming tools, the Outflank - C2 Tool Collection. New tools are released internally first for the OST community. This repository contains a collection of tools which integrate with Cobalt Strike (and possibly other C2 frameworks) through BOF, reflective DLL loading techniques, and .NET assemblies.

Ivanti Connect Secure VPN Privilege Escalation

This is a Cobalt Strike (CS) / Stage1 / Beacon Object File (BOF) which exploits Ivanti Secure Access (previously Pulse Secure) VPN client (CVE-2023-35080). From a low-privileged user mode, it exploits a vulnerability in the kernel driver to overwrite the beacon process token privileges with a system process token privileges. The kernel driver symbolic link has an ACL defining that Everyone can open a handle to it. The research that led to finding this vulnerability has been presented at RedTreat 2023 and BSIDES London 2023.

PowerShell Tradecraft

PowerShell Tradecraft is a collection of tools that support local and remote PowerShell execution. Remote PowerShell execution is performed by lesser known techniques for lateral movement that PowerShell can leverage, tunneling over native Windows techniques. It can be used in Stage 1 C2, Cobalt Strike and there is also a stand alone version for on-host cli usage.

Keyper

Keyper is a small keylogger written in .NET made for usage within your C2 implant (or standalone). It has its use case in specific scenarios despite it not being in BOF format.

Due to the rapid pace of updates to Outflank Security Tooling, this document may not fully represent the latest releases. For the most recent updates, please visit our OST Releases site - https://www.outflank.nl/services/outflank-security-tooling/releases/

Reference OST architectural diagram