# TTP Detail – T1055.001

## TTP Information

Name: Dynamic-link Library Injection

Description: Adversaries may inject dynamic-link libraries (DLLs) into processes in order to evade process-based defenses as well as possibly elevate privileges. DLL injection is a method of executing arbitrary code in the address space of a separate live process.

DLL injection is commonly performed by writing the path to a DLL in the virtual address space of the target process before loading the DLL by invoking a new thread. The write can be performed with native Windows API calls such as <code>VirtualAllocEx</code> and <code>WriteProcessMemory</code>, then invoked with <code>CreateRemoteThread</code> (which calls the <code>LoadLibrary</code> API responsible for loading the DLL). (Citation: Elastic Process Injection July 2017)

Variations of this method such as reflective DLL injection (writing a self-mapping DLL into a process) and memory module (map DLL when writing into process) overcome the address relocation issue as well as the additional APIs to invoke execution (since these methods load and execute the files in memory by manually preforming the function of <code>LoadLibrary</code>).(Citation: Elastic HuntingNMemory June 2017)(Citation: Elastic Process Injection July 2017)

Another variation of this method, often referred to as Module Stomping/Overloading or DLL Hollowing, may be leveraged to conceal injected code within a process. This method involves loading a legitimate DLL into a remote process then manually overwriting the module's <code>AddressOfEntryPoint</code> before starting a new thread in the target process.(Citation: Module Stomping for Shellcode Injection) This variation allows attackers to hide malicious injected code by potentially backing its execution with a legitimate DLL file on disk.(Citation: Hiding Malicious Code with Module Stomping)

Running code in the context of another process may allow access to the process's memory, system/network resources, and possibly elevated privileges. Execution via DLL injection may also evade detection from security products since the execution is masked under a legitimate process.

## Threat-Mapped Scoring

Score: 1.8

Priority: P4 - Informational (Low)

## Kill Chain Phases

**•** mitre-attack: defense-evasion

**•** mitre-attack: privilege-escalation

## Malware

* Aria-body
* BADHATCH
* BlackEnergy
* Bumblebee
* Carberp
* Carbon
* Cobalt Strike
* ComRAT
* Conti
* DarkTortilla
* Derusbi
* Duqu
* Dyre
* Elise
* Emissary
* Emotet
* FinFisher
* FunnyDream
* Gelsemium
* Get2
* HIDEDRV
* Heyoka Backdoor
* Kazuar
* Lizar
* Matryoshka
* Maze
* MegaCortex
* Metamorfo
* Mongall
* Netwalker
* PS1
* PipeMon
* PoisonIvy
* RARSTONE
* RATANKBA
* Ramsay
* Remsec
* SDBbot
* Sagerunex
* Saint Bot
* ShadowPad
* Socksbot
* SombRAT
* Stuxnet
* Sykipot
* Taidoor
* TajMahal
* Uroburos
* ZxShell

## Tools

* IronNetInjector
* Koadic
* PowerSploit
* Pupy

## APTs (Intrusion Sets)

* BackdoorDiplomacy
* Lazarus Group
* Leviathan
* Malteiro
* Putter Panda
* TA505
* Tropic Trooper
* Turla
* Wizard Spider