

# An In-memory Embedding of CPython for Offensive Use

Artifacts

## Abstract

This document describes the steps necessary to download a harness-exe that performs the demonstrations listed in the Demonstration section of the submission “*An In-memory Embedding of CPython for Offensive Use*.” Depending on the specific demonstration, the harness either creates a notepad.exe child process or uses the current console process. The harness-exe process allocates some Write+Execute memory, copies some shellcode and a harness DLL and starts a thread to execute the shellcode. The shellcode loads the harness DLL, and this downloads a CPython DLL constructed as described in the paper, and a demonstration chosen by the user in the steps of harness-exe. The specific steps involving the shellcode and the harness DLL emulate what would happen after an 0-day or N-day exploit without using an actual 0-day or N-day or compromising a system. Because this harness-exe is downloaded from the Internet, it will have a “Mark-of-the-Web,” and Windows Defender will quarantine when you run it. Therefore, please follow the instructions in this document to Add an Exclusion for the harness-exe in Windows Defender.

# Table of Contents

<b>Artifact Checklist</b>	<b>3</b>
<b>Description</b>	<b>3</b>
How to access	3
Installation	4
Experiment workflow	6
Evaluation and expected results	7
Experiment Customization	7

## Artifact Checklist

- Binary: harness-exe.exe
- Run-time environment: verified on Windows 10 x64
- Hardware: A commodity PC is sufficient
- How much disk space required approximately: 100GB
- How much time is needed to prepare workflow: 60 minutes
- How much time is needed to complete experiments: 30 minutes
- Publicly available: Yes, near camera-ready time; now early-release for Artifact reviewers
- Code licenses:

Copyright (C) 2021 SCYTHE, Inc.

Authors: Ateeq Sharfuddin, Brian Chapman, Chris Balles

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

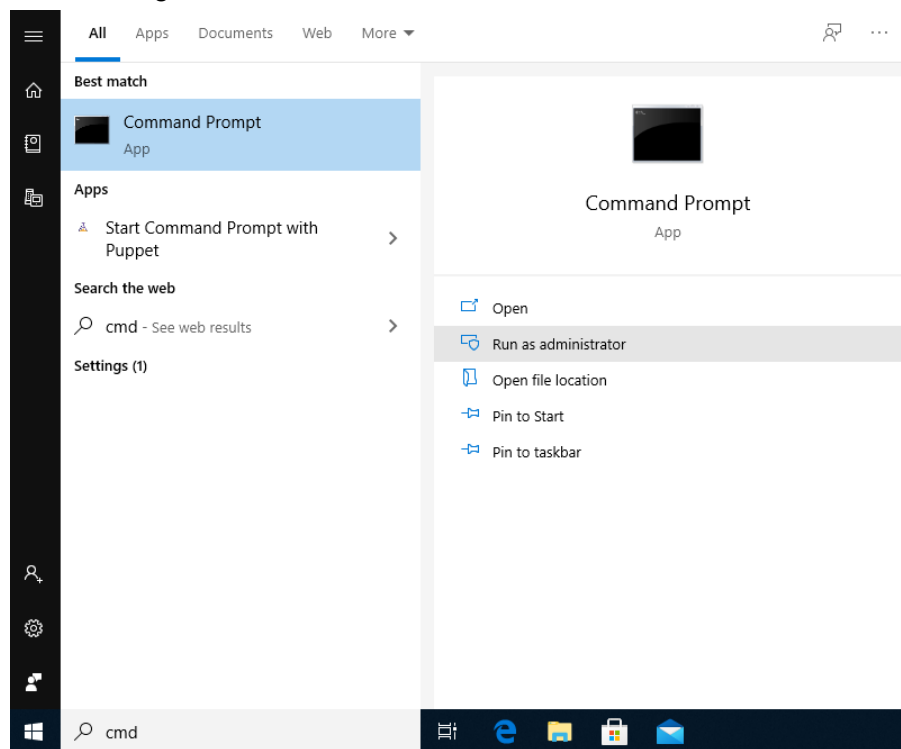
## Description

### How to access

The harness-exe has been verified to work on Microsoft's Windows 10 x64 Edge VM for VMWare. These instructions assume using Microsoft's Edge VM. Step 6 of the Installation section covers how to retrieve and execute harness-exe.

## Installation

1. Download **VMWare Workstation Player** (approximately 215MB) and install on your computer (free for non-commercial use):  
<https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html>
2. Download **MSEdge on Win 10 (x64) Stable 1809** (approximately 6.7GB) from:  
<https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/>. Choose VMWare for the VM platform.
3. Extract the downloaded MSEdge.Win10.VMware.zip and open the **MSEdge-Win10-VMware.ovf** inside the extracted folder MSEdge-Win10-VMware.
4. This will start VMWare Workstation Player importing the Edge Virtual Machine (VM).
5. Log into this Edge VM using:
  - a. User: **IEUser**
  - b. Password: **Passw0rd!**
6. In this VM, create a folder named **woot2021** on the C drive.
7. Start an **Administrative Command Prompt**.
  - a. From the Start menu, type **cmd**, and when Command Prompt appears, choose *Run as administrator* on the right as shown in the figure below and approve the consent dialog box.



## FOR WOOT '21 REVIEWERS

8. Type **powershell** and hit Enter to start a Powershell prompt:

```
Administrator: Command Prompt - powershell
Microsoft Windows [Version 10.0.17763.379]
(c) 2018 Microsoft Corporation. All rights reserved.

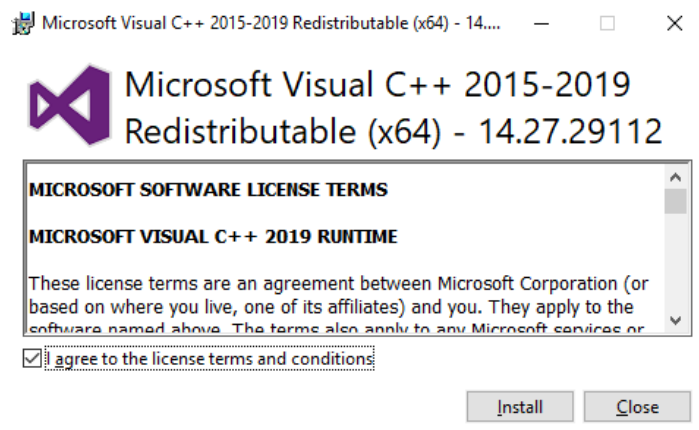
C:\Windows\system32>powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32>
```

9. **Add C:\woot2021 to the Windows Defender Exclusions, turn off automatic sample submission, and download and install Visual Studio redistributable** (approximately 1MB). The installation of the redistributable is a requirement for CPython itself. To accomplish these tasks, copy-and-paste the following commands into the Powershell prompt (Step 5) and hit Enter:

```
mkdir c:\woot2021
cd c:\woot2021
Set-MpPreference -ExclusionPath c:\woot2021
Set-MpPreference -SubmitSamplesConsent NeverSend
Invoke-WebRequest -Uri https://aka.ms/vs/16/release/vc_redist.x64.exe -OutFile "vc_redist.x64.exe"
.\vc_redist.x64.exe
```

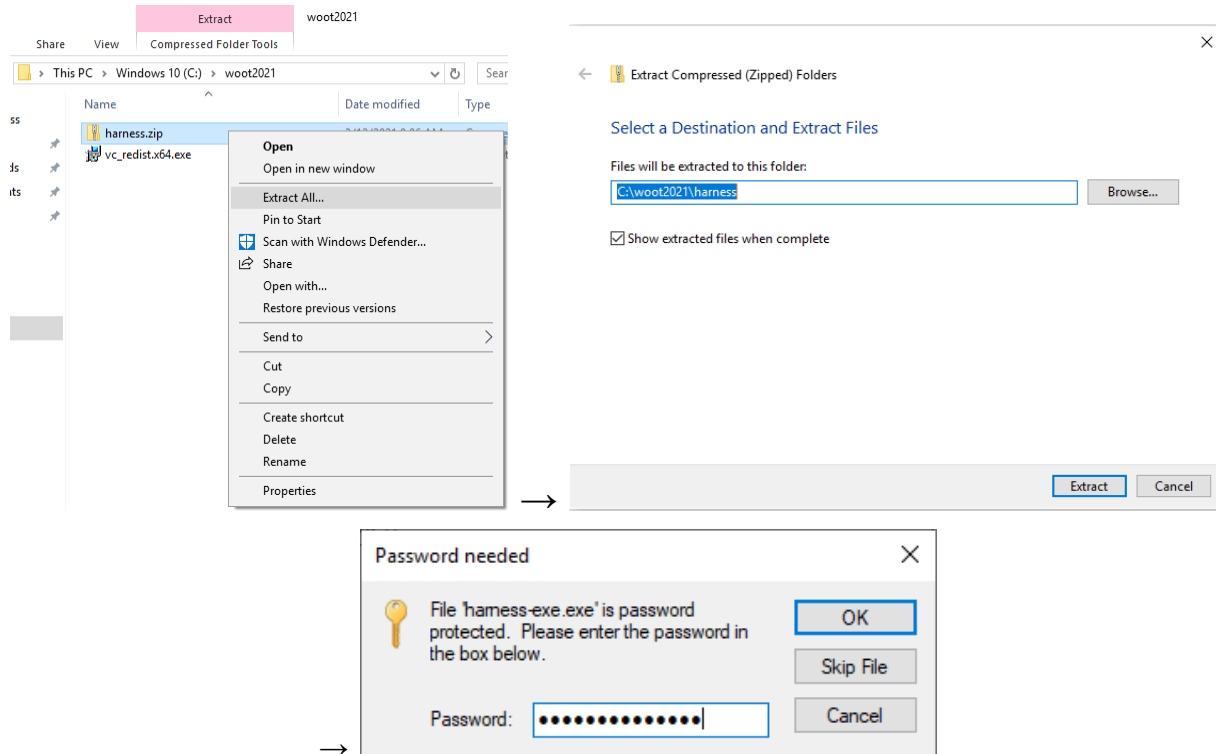
10. Install the redistributable as shown below.



11. Download the **harness-exe** zip also using Powershell and open the folder. Type the following and hit Enter.

```
Invoke-WebRequest -Uri "https://github.com/farfella/woot2021/raw/master/harness/harness-exe.zip" -OutFile "C:\woot2021\harness.zip"
explorer .
```

12. Extract the archive into the C:\woot2021\harness folder. The password to extract is:  
**w00t-2021-w00t**



## Experiment workflow

In the same command prompt as Installation steps, change directory to c:\woot2021\harness. Then run .\harness-exe.exe and hit Enter.

```
CD C:\woot2021\harness
.\harness-exe.exe
```

When you run harness-exe.exe, some instructions are presented. After this, the reviewer may choose the default URL in which the in-memory CPython DLL resides (e.g., <https://github.com/farfella/woot2021/tree/master/in-memory-embedding-cpython>) or download from this GitHub repository and offer the DLL from the reviewer's own hosted server, for example.

Following this step, the reviewer is prompted to choose one of five demonstration options. Options one through four are the examples described in the Demonstrations section in the paper:

1. A demonstration covering all topics (spawns child notepad.exe and shows a dialog box)
2. A demonstration that prints all the process names running on the system

3. A demonstration that downloads a file from the Internet
4. A demonstration calling BCrypt to encrypt and decrypt a string

You will need to run harness-exe.exe for each demonstration.

## Evaluation and expected results

For option 1, a child notepad.exe will be spawned, in which Write+Execute memory will be allocated, and shellcode and harness DLL will be written. Then a thread will be spawned in this child notepad.exe to execute this shellcode. The shellcode will load the harness DLL. The harness DLL will retrieve the CPython DLL constructed as described in the paper and also the zip file corresponding to demonstration 1 from either the default URLs or reviewer-provided URLs.

For option 2, 3, and 4, in the current process (i.e., process of harness-exe.exe) the write+execute memory will be allocated and the shellcode and harness DLL will be written. This is because these examples use print() to write to standard output, and the current process is a console application that can support this functionality. The CPython DLL is instantiated with verbosity turned on (i.e., equivalent to running python -v). As such, you will see debug messages on the screen to give you more insight.

## Experiment Customization

The fifth demonstration option allows the reviewer to offer a zip file with a module named magic inside (i.e., magic.py or magic.pyc) from their own hosted URL. The harness DLL is constructed to only 'import magic'. This allows the reviewer to test with some custom Python code instead of the four demonstration options. Note however, that only the standard Python packages are embedded in this CPython DLL. We stand ready to assist if you want to customize.