Script Monitors and Discoveries

Ing. Ondřej Ševeček | GOPAS a.s. |
MCM:Directory | MVP:Enterprise Security | CEH:Certified Ethical Hacker | CHFI: Computer Hacking Forensic Investigator
ondrej@sevecek.com | www.sevecek.com |

GOPAS: info@gopas,cz | www.gopas.cz | www.facebook.com/P.S.GOPAS

Replacement elements

```
# Parameter values, just a plain string replacement done by the Agent
# for example: you can use it inside WMI queries as well
$Target/Property[Type='<class>']/<property>$
$Target/Host/Host/Property[Type='<class>']/<property>$

# targetId for disco script
$Target/Id$

# sourceId for disco script
$MPElement$

# Inside disco script, the class and its properties
$MPElement[Name='<class>']$
$MPElement[Name='<class>']/<property>$

$MPElement[Name='System!System.Entity']/DisplayName$
$MPElement[Name='Windows!Microsoft.Windows.Computer']/PrincipalName$
```

Monitor and property bags

```
param(
     )

$scom = New-Object -Com 'MOM.ScriptAPI'
$scom.LogScriptEvent('Sevecek script', 19001, 0, "Script message")

$bag = $scom.CreatePropertyBag()


$bag.AddValue('StrictConsistency', $strictConsistency)
$bag.AddValue('CorruptPartner', $corruptPartner)


return $bag
```

GOPAS

Monitor type

```
<ModuleTypes>

<DataSourceModuleType ID="Sevecek.Monitoring.TimedPowerShell.PropertyBagProvider" >
<ModuleImplementation>
  <Composite>
    <MemberModules>
      <DataSource TypeID="System!System.SimpleScheduler">
        <IntervalSeconds>$Config/IntervalSeconds$</IntervalSeconds>
      <ProbeAction
TypeID="Windows!Microsoft.Windows.PowerShellPropertyBagTriggerOnlyProbe" >
  <OutputType>System!System.PropertyBagData</OutputType>
</ModuleTypes>


<MonitorTypes>

<UnitMonitorType ID="Sevecek.Monitoring.TimedPowerShell.ThreeStateMonitorType" >
  <MonitorImplementation>
    <MemberModules>
      <DataSource TypeID="Sevecek.Monitoring.TimedPowerShell.PropertyBagProvider" >

</MonitorTypes>
```

GOPAS

Discovery

```
param(
     [string] $sourceID,
     [string] $targetID,
     [string] $computer
     )

$scom = New-Object -Com 'MOM.ScriptAPI'
$discoData = $scom.CreateDiscoveryData(0, $sourceID, $targetID)

$oneObject = $discoData.CreateClassInstance('$MPElement[Name="<class>"]$')

$oneObject.AddProperty('$MPElement[Name="System!System.Entity"]/DisplayName$'
, '<display>')

$oneObject.AddProperty('$MPElement[Name="Windows!Microsoft.Windows.Computer"]
/PrincipalName$', $computer)

$oneObject.AddProperty('$MPElement[Name="<class>"]/<keyProperty>$', '<key>')
$oneObject.AddProperty('$MPElement[Name="<class>"]/<other>$', '<other>')

$discoData.AddInstance($oneObject)

return $discoData
```

GOPAS

Example monitoring

- Monitor insecure registry on DCs
  - HKLM\SYSTEM\CurrentControlSet\Services\NTDS\Parameters
  - Allow Replication with Divergent and Corrupt Partner = 1
  - Strict Replication Consistency = 0
  - Repl Perform Initial Synchronizations = 0
- !sevecek-datadisk-*.txt
- Win32_LogicalDisk FreeSpace with WMI performance monitor hosted on a new "DataDisk" object

GOPAS