### ConnectionLogger

ConnectionLogger is a console based x86 (32-bit) programs that runs on all versions of Windows from Windows XP to Windows 2016. The utility was written by Ansar Mohammed of MCS Canada. The source code is available internally within Microsoft.

The implementation document discusses in detail on how to use the ConnectionLogger.

The primary benefits of using ConnectionLogger are

* Minimal executable footprint on the target server
* Minimal logging footprint on the target server
* Specific targeted logging
* Extensible to support multiple protocols
* It performs much like a sniffer without the need to install the drivers required for host based sniffing or the network disruption required for a hardware sniffer

#### Choosing Ports

The following is the recommended list of ports to monitor for ConnectionLogger

|  |  |  |
| --- | --- | --- |
| Protocol | Port | Reason |
| TCP | 53 | * Identify any undocumented DNS servers are requesting Zone transfers |
| TCP | 88 | * Identify any statically configured Kerberos clients |
| TCP | 389 | * Identify any statically configured LDAP clients |
| TCP | 636 | * Identify any statically configured LDAPs clients |
| TCP | 3268 | * Identify any statically configured GC clients |
| TCP | 3269 | * Identify any statically configured GCs clients |
| UDP | 53 | * Identify any statically configured DNS resolvers |
| UDP | 123 | * Identify statically configured NTP clients |
| UDP | 389 | * Identify any statically configured LDAP clients |

#### Common Problems

* ConnectionLogger requires local administrator rights. If running on a server with UAC enabled, you must run it on an elevated command prompt. Further, some customers have expressed interest in running it as a service.
* If the customer runs a UDP scanner against a host running the ConnectionLogger, it will list all services that the UDP scanner was targeting in the connectionlogger.csv file.
* If the customer runs a TCP scanner against a host running the ConnectionLogger, it will only list the services that where connections were made in the connectionlogger.csv file.
* The Windows firewall must have an exception for the ConnectionLogger executable if the firewall is enabled.

#### Testing ConnectionLogger

The following PowerShell script can be used to test ConnectionLogger.

$tcpPortList = @(53,88,139,389,445,636,3268,3269)

$udpPortList = @(53,88,123,389)

$TargetDC = "LEDC1"

function Start-TCPPortConnect {

Param (

[Parameter(Position=0,

Mandatory=$true,

ValueFromPipeline=$True,

ValueFromPipelineByPropertyName=$True)]

[string]$RemoteHost,

[Parameter(Position=1,

Mandatory=$true,

ValueFromPipeline=$True,

ValueFromPipelineByPropertyName=$True)]

[string]$Port)

$tcp = new-object net.sockets.tcpclient

try

{

$tcp.connect($remotehost, $port)

Write-Host "Connected to port " $port " on " $remotehost

}

catch [Exception]

{

Write-host $\_.Exception.Message

}

}

function Start-UDPPortConnect {

Param (

[Parameter(Position=0,

Mandatory=$true,

ValueFromPipeline=$True,

ValueFromPipelineByPropertyName=$True)]

[string]$RemoteHost,

[Parameter(Position=1,

Mandatory=$true,

ValueFromPipeline=$True,

ValueFromPipelineByPropertyName=$True)]

[string]$Port)

$udp = new-object net.sockets.udpclient

try

{

$udp.connect($remotehost, $port)

$ret = $udp.Send(1,1)

Write-Host "Packet on port" $port " sent to " $remotehost

}

catch [Exception]

{

Write-host $\_.Exception.Message

}

}

$udpPortList | ForEach-Object {

Start-UDPPortConnect $TargetDC $\_

}

$tcpPortList | ForEach-Object {

Start-TCPPortConnect $TargetDC $\_

}

#### Interpretation of Results

ConnectionLogger outputs to a ***connectionlogger.db*** SQLite file. This file is parsed with the companion exporter executable. The exporter can be run while collection is ongoing.

Standard output looks as follows:

SourceIPAddress, DestinationProtocol, DestinationPort, TimeStamp

208.84.2.53, udp, 59460, Thu May 01 03:41:27 2014

208.84.2.53, udp, 57631, Thu May 01 03:41:27 2014

85.17.73.194, tcp, 58351, Thu May 01 03:41:27 2014

The top line of the csv labels the columns. Each line in the csv represents a “flow” of traffic from a:

* SourceIPAddress: The dotted decimal notation of the source IP Address making the connection to the server
* DestinationProtocol: either TCP or UDP
* DestinationPort: The service that the SourceIPAddress accessed
* TimeStamp: the last time that the ConnectionLogger saw the traffic.

ConnectionLogger does not record all sessions. It only records the last connection. Future releases may include a count of the number of sessions or packets. This would add significant overhead to the processing of packets, especially in the processing of UDP packets as there is no way of tracking UDP sessions without deep packet inspection.