Windows Filtering Platform (WFP) Sampler

The WFPSampler is a fully functional firewall comprised of the following binaries:

❖  WFPSampler.exe

❖  WFPSamplerService.Exe

❖  WFPSamplerCalloutDriver.Sys

❖  WFPSamplerProxyService.Exe

**WFPSampler.Exe**

**Usage:**

This executable is the command-line interface used by the user to define the policy.

“**WFPSampler.exe -?**” provides output of valid parameters to pass.

“**WFPSampler.exe -s <SCENARIO> -?**” provides output of valid parameters pertaining to that scenario.

**Source:**

The executable is located under exe and is comprised of the following files:

❖  Framework\_RPCClientInterface.CPP

❖  Framework\_RPCClientInterface.H

❖  Framework\_WFPSampler.CPP

❖  Framework\_WFPSampler.H

❖  Framework\_WFPSampler.rc

❖  HelperFunctions\_CommandLine.CPP

❖  HelperFunctions\_CommandLine.H

❖  Scenarios\_AppContainers.CPP

❖  Scenarios\_AppContainers.H

❖  Scenarios\_BasicAction.CPP

❖  Scenarios\_BasicAction.H

❖  Scenarios\_BasicPacketExamination.CPP

❖  Scenarios\_BasicPacketExamination.H

❖  Scenarios\_BasicPacketInjection.CPP

❖  Scenarios\_BasicPacketInjection.H

❖  Scenarios\_BasicPacketModification.CPP

❖  Scenarios\_BasicPacketModification.H

❖  Scenarios\_BasicStreamInjection.CPP

❖  Scenarios\_BasicStreamInjection.H

❖  Scenarios\_FastPacketInjection.CPP

❖  Scenarios\_FastPacketInjection.H

❖  Scenarios\_FastStreamInjection.CPP

❖  Scenarios\_FastStreamInjection.H

❖  Scenarios\_FlowAssociation.CPP

❖  Scenarios\_FlowAssociation.H

❖  Scenarios\_Include.H

❖  Scenarios\_PendAuthorization.CPP

❖  Scenarios\_PendAuthorization.H

❖  Scenarios\_PendEndpointClosure.CPP

❖  Scenarios\_PendEndpointClosure.H

❖  Scenarios\_Proxy.CPP

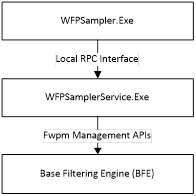
❖  Scenarios\_Proxy.H

and links with lib\WFPSampler.lib.

The **Framework\_\*** files contain functions for creating and maintaining the executable such as wmain.  These are relatively safe to ignore if you are only concerned with WFP functionality.

The **HelperFunctions\_CommandLine.\*** are files which assist with parsing of the command line parameters passed.  This has the building blocks for the FWPM\_FILTER and FWPM\_PROVIDER\_CONTEXT that the WFPSamplerService.Exe uses for defining policy.

The **Scenarios\_\*** are files which gather and package all the data required by the WFPSamplerService.Exe.  The packaged data is sent to the WFPSamplerService.Exe via local RPC interfaces exposed by the service.



**Figure A.   Interaction between modules**

**WFPSamplerService.Exe**

**Usage:**

This is the service which instructs BFE to add or remove policies.

“**WFPSamplerService.Exe -i**“ installs the service.  While installed, the service is under the control of the Service Control Manager (as WFPSampler).  The service is set to autostart and depends on BFE.

“**WFPSamplerService.Exe -u**“ uninstalls the service.

**Source:**

The executable is located under svc and is comprised of the following files:

❖  Framework\_ActionCenter.CPP \*

❖  Framework\_ActionCenter.H \*

❖  Framework\_Include.H

❖  Framework\_RPCServerInterface.CPP

❖  Framework\_RPCServerInterface.H

❖  Framework\_WFPSamplerService.CPP

❖  Framework\_WFPSamplerService.H

❖  Framework\_WFPSamplerService.RC

❖  Framework\_WFPSamplerService\_Msg.MC

❖  Framework\_WindowsFirewall.CPP

❖  Framework\_WindowsFirewall.H

❖  Scenarios\_AppContainers.CPP

❖  Scenarios\_BasicAction.CPP

❖  Scenarios\_BasicPacketExamination.CPP

❖  Scenarios\_BasicPacketInjection.CPP

❖  Scenarios\_BasicPacketModification.CPP

❖  Scenarios\_BasicStreamInjection.CPP

❖  Scenarios\_FastPacketInjection.CPP

❖  Scenarios\_FastStreamInjection.CPP

❖  Scenarios\_FlowAssociation.CPP

❖  Scenarios\_PendAuthorization.CPP

❖  Scenarios\_PendEndpointClosure.CPP

❖  Scenarios\_Proxy.CPP

\*Available only under Non Disclosure Agreement (NDA)

and links with lib\WFPSampler.lib.

The **Framework\_\*** files contain functions for creating and maintaining the service such as ServiceMain.  These are relatively safe to ignore if you are only concerned with WFP functionality with the following exceptions:  **Framework\_WindowsFirewall.CPP** demonstrates how to properly disable Microsoft Windows Firewall, and if you have a signed NDA agreement, **Framework\_ActionCenter.CPP** demonstrates how to register with Microsoft Windows Action Center.

The **Scenarios\_\*** files finish creating the filters and provider contexts required by each scenario.  These are then added to the system using Fwpm Management calls.

**WFPSamplerCalloutDriver.Sys**

**Usage:**

This is the driver which houses the various callout functions.

**Source:**

This driver is located under sys and is comprised of the following files:

❖  ClassifyFunctions\_BasicActionCallouts.CPP

❖  ClassifyFunctions\_BasicActionCallouts.H

❖  ClassifyFunctions\_BasicPacketExaminationCallouts.CPP

❖  ClassifyFunctions\_BasicPacketExaminationCallouts.H

❖  ClassifyFunctions\_BasicPacketInjectionCallouts.CPP

❖  ClassifyFunctions\_BasicPacketInjectionCallouts.H

❖  ClassifyFunctions\_BasicPacketModificationCallouts.CPP

❖  ClassifyFunctions\_BasicPacketModificationCallouts.H

❖  ClassifyFunctions\_BasicStreamInjectionCallouts.CPP

❖  ClassifyFunctions\_BasicStreamInjectionCallouts.H

❖  ClassifyFunctions\_FastPacketInjectionCallouts.CPP

❖  ClassifyFunctions\_FastPacketInjectionCallouts.H

❖  ClassifyFunctions\_FastStreamInjectionCallouts.CPP

❖  ClassifyFunctions\_FastStreamInjectionCallouts.H

❖  ClassifyFunctions\_Include.H

❖  ClassifyFunctions\_PendAuthorizationCallouts.CPP

❖  ClassifyFunctions\_PendAuthorizationCallouts.H

❖  ClassifyFunctions\_PendEndpointClosureCallouts.CPP

❖  ClassifyFunctions\_PendEndpointClosureCallouts.H

❖  ClassifyFunctions\_ProxyCallouts.CPP

❖  ClassifyFunctions\_ProxyCallouts.H

❖  CompletionFunctions\_BasicPacketInjectionCallouts.CPP

❖  CompletionFunctions\_BasicPacketInjectionCallouts.H

❖  CompletionFunctions\_BasicPacketModificationCallouts.CPP

❖  CompletionFunctions\_BasicPacketModificationCallouts.H

❖  CompletionFunctions\_BasicStreamInjectionCallouts.CPP

❖  CompletionFunctions\_BasicStreamInjectionCallouts.H

❖  CompletionFunctions\_FastPacketInjectionCallouts.CPP

❖  CompletionFunctions\_FastPacketInjectionCallouts.H

❖  CompletionFunctions\_FastStreamInjectionCallouts.CPP

❖  CompletionFunctions\_FastStreamInjectionCallouts.H

❖  CompletionFunctions\_Include.H

❖  CompletionFunctions\_PendAuthorizationCallouts.CPP

❖  CompletionFunctions\_PendAuthorizationCallouts.H

❖  CompletionFunctions\_ProxyCallouts.CPP

❖  CompletionFunctions\_ProxyCallouts.H

❖  Framework\_Events.CPP

❖  Framework\_Events.H

❖  Framework\_Include.H

❖  Framework\_PowerStates.CPP

❖  Framework\_PowerStates.H

❖  Framework\_WFPSamplerCalloutDriver.CPP

❖  Framework\_WFPSamplerCalloutDriver.H

❖  Framework\_WFPSamplerCalloutDriver.rc

❖  HelperFunctions\_ExposedCallouts.CPP

❖  HelperFunctions\_ExposedCallouts.H

❖  NotifyFunctions\_BasicCallouts.CPP

❖  NotifyFunctions\_BasicCallouts.H

❖  NotifyFunctions\_FastCallouts.CPP

❖  NotifyFunctions\_FastCallouts.H

❖  NotifyFunctions\_FlowDelete.CPP

❖  NotifyFunctions\_FlowDelete.H

❖  NotifyFunctions\_Include.H

❖  NotifyFunctions\_PendCallouts.CPP

❖  NotifyFunctions\_PendCallouts.H

❖  NotifyFunctions\_ProxyCallouts.CPP

❖  NotifyFunctions\_ProxyCallouts.H

❖  SubscriptionFunctions\_BFEState.CPP

❖  SubscriptionFunctions\_BFEState.H

❖  SubscriptionFunctions\_Include.H

❖  WFPSamplerCalloutDriver.InX

and links with SysLib\WFPSamplerSys.Lib.

The **Framework\_\*** files contain functions for creating and maintaining a filter driver using the Microsoft Windows Driver Foundation.  These are relatively safe to ignore if you are only concerned with WFP functionality.

**HelperFunctions\_ExposedCallouts.CPP** contains functions used to register and unregister the various callouts exposed by this driver.  While the methodology of creating the array of the various callouts is inconsequential,   it does demonstrate proper use of filling out the FWPS\_CALLOUT object and calling FwpsCalloutRegister and FwpsCalloutUnregisterByKey.

The **ClassifyFunctions\_\***are files which contain the various FWPS\_CALLOUT\_CLASSIFY\_FN and the work they perform.  The classifyFns are all laid out the same:

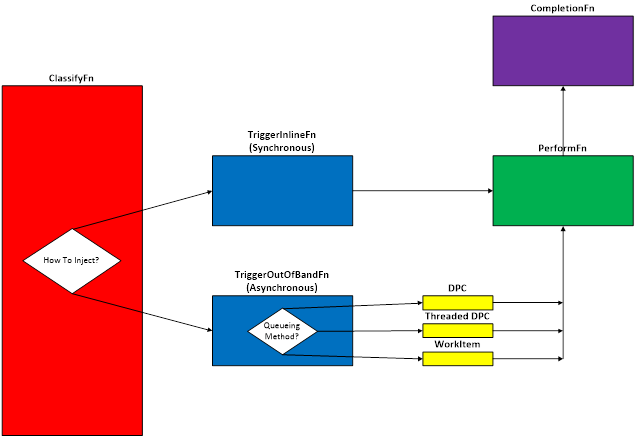
•         **Classify\*** functions are the initial classify.  This function will see if any further processing is needed, and if so, which Trigger function to call (inline (synchronous) or out of band (asynchronous)).

•         **Trigger\*** are functions which packages the data of the initial classify and either send it to the appropriate Perform function or to the WorkItemRoutine, DeferredProcedureCall, or TDeferredProcedureCall.

•         **\*WorkItemRoutine** are functions which finish packaging the necessary data, and queues a WorkItem for asynchronous processing at PASSIVE\_LEVEL.

•         **\*DeferredProcedureCall** are functions which finish packaging the necessary data, and queuesa DPC for asynchronous processing at PASSIVE\_LEVEL.  This is the default out of band method.

•         **\*Perform** are functions which do the actual processing desired by the callout.

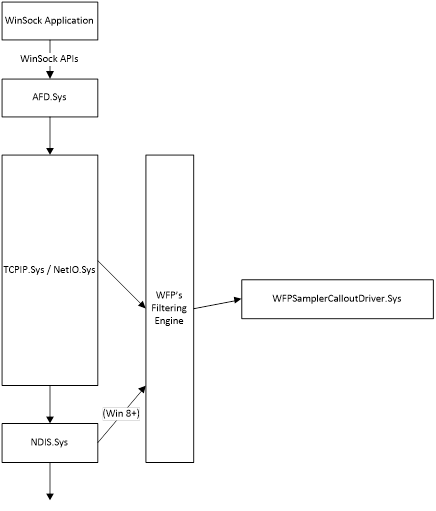


The **CompletionFunctions\_\*** are files which contain the various FWPS\_INJECT\_COMPLETE.

The **NotifyFunctions\_\*** are files which contain the various FWPS\_CALLOUT\_NOTIFY\_FN and FWPS\_CALLOUT\_FLOW\_DELETE\_NOTIFY\_FN.

The **SubscriptionFunctions\_\*** are files which contain the various callbacks used by subscriptions.

**WFPSamplerCalloutDriver.InX** is a file used in creation of the Inf used to install the driver.



**Figure B.   Packet Traversal**

**WFPSamplerProxyService.Exe**

**Usage:**

This is the service which listens for connections to proxy.

“**WFPSamplerProxyService.Exe -i**“  will install the service.  While installed, the service is under the control of the Service Control Manager (as WFPSamplerProxy).  The service is set to autostart.

“**WFPSamplerProxyService.Exe -u**“ will uninstall the service.

**Source:**

The executable is located under ProxySvc and is comprised of the following files:

❖  Framework\_ProxySvc.CPP

❖  Framework\_ProxySvc.H

❖  Framework\_ProxySvc.RC

❖  Framework\_ProxySvc\_Msg.MC

❖  ThreadRoutines\_Include.H

❖  ThreadRoutines\_TCPListener.CPP

❖  ThreadRoutines\_TCPListener.H

❖  ThreadRoutines\_UDPListener.CPP

❖  ThreadRoutines\_UDPListener.H

The **Framework\_\*** files contain functions for creating and maintaining the service such as ServiceMain.  These are relatively safe to ignore if you are only concerned with WFP functionality.

The **ThreadRoutines\_\***files contain functions for maintaining the proxy.  These are relatively safe to ignore if you are only concerned with WFP functionality.

**WFPSampler.Lib**

**Usage:**

This is a library of user mode helper functions used throughout the project.

**Source:**

This library is located under Lib and is comprised of the following files:

❖  HelperFunctions\_FwpmCallout.CPP

❖  HelperFunctions\_FwpmCallout.H

❖  HelperFunctions\_FwpmEngine.CPP

❖  HelperFunctions\_FwpmEngine.H

❖  HelperFunctions\_FwpmFilter.CPP

❖  HelperFunctions\_FwpmFilter.H

❖  HelperFunctions\_FwpmLayer.CPP

❖  HelperFunctions\_FwpmLayer.H

❖  HelperFunctions\_FwpmProvider.CPP

❖  HelperFunctions\_FwpmProvider.H

❖  HelperFunctions\_FwpmProviderContext.CPP

❖  HelperFunctions\_FwpmProviderContext.H

❖  HelperFunctions\_FwpmSubLayer.CPP

❖  HelperFunctions\_FwpmSubLayer.H

❖  HelperFunctions\_FwpmTransaction.CPP

❖  HelperFunctions\_FwpmTransaction.H

❖  HelperFunctions\_GUID.CPP

❖  HelperFunctions\_GUID.H

❖  HelperFunctions\_Include.H

❖  HelperFunctions\_IPAddress.CPP

❖  HelperFunctions\_IPAddress.H

❖  HelperFunctions\_Log.CPP

❖  HelperFunctions\_Log.H

❖  HelperFunctions\_MACAddress.CPP

❖  HelperFunctions\_MACAddress.H

❖  HelperFunctions\_Macros.H

❖  HelperFunctions\_Process.CPP

❖  HelperFunctions\_Process.H

❖  HelperFunctions\_Registry.CPP

❖  HelperFunctions\_Registry.H

❖  HelperFunctions\_Service.CPP

❖  HelperFunctions\_Service.H

❖  HelperFunctions\_SID.CPP

❖  HelperFunctions\_SID.H

❖  HelperFunctions\_Strings.CPP

❖  HelperFunctions\_Strings.H

❖  HelperFunctions\_ThreadPools.CPP

❖  HelperFunctions\_ThreadPools.H

❖  HelperFunctions\_ThreadsAndEvents.CPP

❖  HelperFunctions\_ThreadsAndEvents.H

❖  HelperFunctions\_WinSock.CPP

❖  HelperFunctions\_WinSock.H

If you are interested only in the WFP functionality, then most attention should be with the HelperFunctions\_Fwpm\* and HelperFunctions\_WinSock.CPP.

**WFPSamplerSys.Lib**

**Usage:**

This is a library of kernel mode helper functions used throughout the project.

**Source:**

This library is located under SysLib and is comprised of the following files:

❖  HelperFunctions\_ClassifyData.CPP

❖  HelperFunctions\_ClassifyData.H

❖  HelperFunctions\_DeferredProcedureCalls.CPP

❖  HelperFunctions\_DeferredProcedureCalls.H

❖  HelperFunctions\_FlowContext.CPP

❖  HelperFunctions\_FlowContext.H

❖  HelperFunctions\_FwpObjects.CPP

❖  HelperFunctions\_FwpObjects.H

❖  HelperFunctions\_Headers.CPP

❖  HelperFunctions\_Headers.H

❖  HelperFunctions\_ICMPMessages.H

❖  HelperFunctions\_Include.H

❖  HelperFunctions\_InjectionData.CPP

❖  HelperFunctions\_InjectionData.H

❖  HelperFunctions\_Macros.H

❖  HelperFunctions\_NDIS.CPP

❖  HelperFunctions\_NDIS.H

❖  HelperFunctions\_NetBuffer.CPP

❖  HelperFunctions\_NetBuffer.H

❖  HelperFunctions\_NotifyData.H

❖  HelperFunctions\_PendData.CPP

❖  HelperFunctions\_PendData.H

❖  HelperFunctions\_RedirectData.CPP

❖  HelperFunctions\_RedirectData.H

❖  HelperFunctions\_WorkItems.CPP

❖  HelperFunctions\_WorkItems.H

If you are interested only in the WFP functionality, then most attention should be with HelperFunctions\_FwpObjects.CPP.

**Miscellaneous**

**HCK**

**Usage:**

This directory contains an answer and info file that can be used in conjunction with the Windows Hardware Certification Kit.  To validate that the WFPSampler is fully WHCK compliant, you can install the WFPSampler on a WHCK client machine, and run the WindowsFilteringPlatform\_Tests.  When prompted by the test to populate the info file, copy these files to %WinDir%\System32\WFPLogo.Answer and %WinDir%\System32\WFPLogo.Info.  Exit out of the WFPLogo.Info that is currently displayed (without saving). Press “OK” to have the test continue.

**Source:**

❖  WFPLogo\_WFPSampler.Answer

❖  WFPLogo\_WFPSampler.Info

**IDL**

**Usage:**

This directory contains files used by the MIDL compiler to create the RPC interface between WFPSampler.Exe and WFPSamplerService.Exe

**Source:**

❖  WFPSamplerRPC.ACF

❖  WFPSamplerRPC.IDL

**Scripts**

**Usage:**

This directory contains scripts used to install and uninstall the WFPSampler.

**Sources:**

❖  WFPSamplerInstall.cmd

“**WFPSamplerInstall.cmd**“ will copy the necessary binaries to their appropriate location, and install each component.

“**WFPSamplerInstall.cmd -r** “ will uninstall each component and remove the binaries from the appropriate location.