ADVANCED PACKET INJECTION

**Overview**

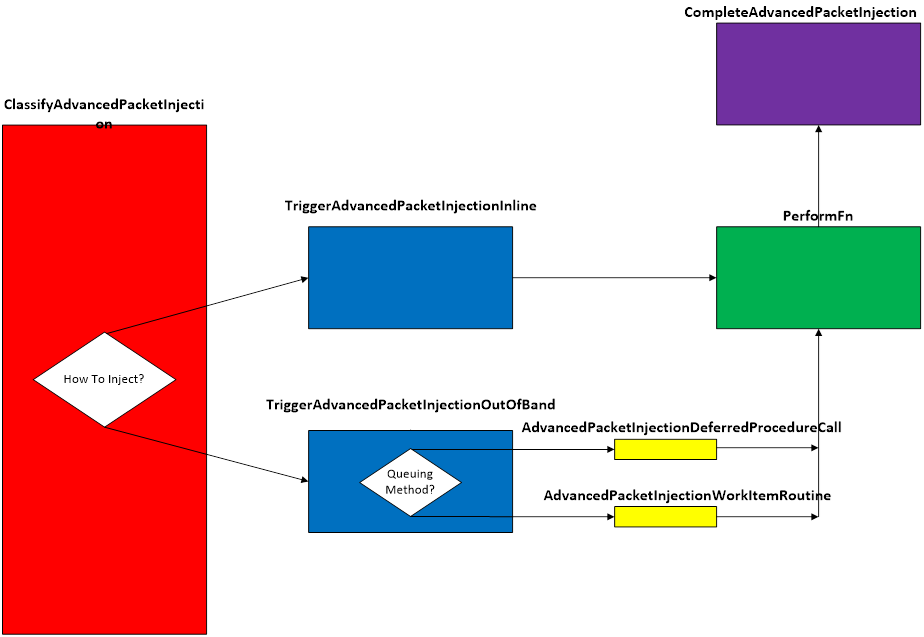
The Advanced Packet Injection scenario will create a new packet based off of the original packet and inject it back to the same layer.  No modification is performed on the packet, with the following exceptions:

•         The packet is loopback and only one of either the source or destination address is that of the software loopback.  In this case, the IP header is modified by changing the software loopback address to that of an address on the local machine.  This is required due to TCPIP’s stack validation logic.

•         The packet is received at an inbound transport layer, and had been IPsec secured. In this case, it is necessary to reconstruct the IP header as IPsec processing does not remove the original ESP or AH information from it.

All filters added sit in WFPSampler's sublayer (which is weighted just below IPsec's sublayer), unless otherwise specified using the -sl <SUBLAYER> command line option.  All filters are associated with WFPSampler's provider.

The following diagram shows how the code flows for this callout:

  
**Figure A. Code flow for Advanced Packet Injection Scenario**

When traffic matches a filter at the specified layer, **ClassifyAdvancedPacketInjection()** is invoked by the Filtering Engine.  This function validates that we can perform the injection by looking at the pClassifyOut rights.  It will then create the INJECTION\_DATA which consists of the injectionHandle and the injectionState.  If the injectionState indicates that we haven’t injected this packet before, then the injection method is determined (default is asynchronous), and the appropriate triggerFn is called.   At this point, the original packet will be blocked.

If the injection method is synchronous (inline), **TriggerAdvancedPacketInjectionInline()** is invoked.  This function creates the CLASSIFY\_DATA, which consists of the data that was passed into the classifyFn.  Depending on which layer the injection is happening, the appropriate performFn is called.

If the injection method is asynchronous (out of band), **TriggerAdvancedPacketInjectionOutOfBand()** is invoked.  This function creates the CLASSIFY\_DATA which consists of copies and references of the data that was passed into the classifyFn.  Based on the queuing method, the appropriate queueFn (default is DeferredProcedureCalls) is invoked.

Regardless of which queueFn is used, each will call the appropriate performFn based on the layer the injection is happening.

Each of the performFns are tailored to inject for their respective layers.  Each will get the required data for its specific injectionFn.  Depending on the layer, the offsets are adjusted on the original so the whole packet is available.  Once the offsets are adjusted, a new NBL is created, the original data is copied over, and the offsets of the original are returned to the original place.  If any of the exceptions noted above are required, then modification of the IP header occurs. Finally, the appropriate injectionFn is invoked.

Upon successful injection, **CompleteAdvancedPacketInjection()** will be called by the TCPIP stack.  This function will show the status of the injected packet.  Additionally, any memory that was allocated from the functions above, will be freed and any references released.

**Applicable Layers**

❖  FWPM\_LAYER\_INBOUND\_IPPACKET\_V4

❖  FWPM\_LAYER\_INBOUND\_IPPACKET\_V6

❖  FWPM\_LAYER\_OUTBOUND\_IPPACKET\_V4

❖  FWPM\_LAYER\_OUTBOUND\_IPPACKET\_V6

❖  FWPM\_LAYER\_IPFORWARD\_V4

❖  FWPM\_LAYER\_IPFORWARD\_V6

❖  FWPM\_LAYER\_INBOUND\_TRANSPORT\_V4

❖  FWPM\_LAYER\_INBOUND\_TRANSPORT\_V6

❖  FWPM\_LAYER\_OUTBOUND\_TRANSPORT\_V4

❖  FWPM\_LAYER\_OUTBOUND\_TRANSPORT\_V6

❖  FWPM\_LAYER\_DATAGRAM\_DATA\_V4

❖  FWPM\_LAYER\_DATAGRAM\_DATA\_V6

❖  FWPM\_LAYER\_INBOUND\_ICMP\_ERROR\_V4

❖  FWPM\_LAYER\_INBOUND\_ICMP\_ERROR\_V6

❖  FWPM\_LAYER\_OUTBOUND\_ICMP\_ERROR\_V4

❖  FWPM\_LAYER\_OUTBOUND\_ICMP\_ERROR\_V6

❖  FWPM\_LAYER\_ALE\_AUTH\_RECV\_ACCEPT\_V4

❖  FWPM\_LAYER\_ALE\_AUTH\_RECV\_ACCEPT\_V6

❖  FWPM\_LAYER\_ALE\_AUTH\_CONNECT\_V4

❖  FWPM\_LAYER\_ALE\_AUTH\_CONNECT\_V6

❖  FWPM\_LAYER\_ALE\_FLOW\_ESTABLISHED\_V4

❖  FWPM\_LAYER\_ALE\_FLOW\_ESTABLISHED\_V6

❖  FWPM\_LAYER\_STREAM\_PACKET\_V4                                     (Win7+)

❖  FWPM\_LAYER\_STREAM\_PACKET\_V6                                     (Win7+)

❖  FWPM\_LAYER\_INBOUND\_MAC\_FRAME\_ETHERNET       (Win8+)

❖  FWPM\_LAYER\_OUTBOUND\_MAC\_FRAME\_ETHERNET   (Win8+)

❖  FWPM\_LAYER\_INBOUND\_MAC\_FRAME\_NATIVE            (Win8+)

❖  FWPM\_LAYER\_OUTBOUND\_MAC\_FRAME\_NATIVE        (Win8+)

❖  FWPM\_LAYER\_INGRESS\_VSWITCH\_ETHERNET                  (Win8+)

❖  FWPM\_LAYER\_EGRESS\_VSWITCH\_ETHERNET                    (Win8+)

**Command Line Usage**

|  |  |  |
| --- | --- | --- |
| **Option** | **Argument** | **Meaning** |
| -s | ADVANCED\_PACKET\_INJECTION | Implement the ADVANCED\_PACKET\_INJECTION scenario |
| -l | Applicable Layer | Layer at which this filter will apply |
| -sl | Applicable subLayer | SubLayer to associate with the filter.  [default is WFPSAMPLER\_SUBLAYER]. |
| -v |  | Make the objects associated with this scenario’s instance dynamic |
| -b |  | Make the objects associated with this scenario’s instance available during boot-time |
| -in |  | Perform the injection synchronously (inline) |
| -tdpc |  | Use threaded DPCs for asynchronous (out of band) queuing method |
| -wi |  | Use work items for asynchronous (out of band) queuing method |
| -r |  | Remove objects associated with this scenario instance |
| -ab | Additional Bytes to allocate | Create a new packet the size of the original + these additional bytes |
| -? |  | Display help |

“**WFPSampler.Exe -s ADVANCED\_PACKET\_INJECTION -?**“ provides help output

“**WFPSampler.Exe -s ADVANCED\_PACKET\_INJECTION -l FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 -v**“ adds a dynamic filter (**-v**) at FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 (**-l**) which references the appropriate callout.  This filter will have no conditions, meaning it will act on all traffic seen at this layer.

“**WFPSampler.Exe -s ADVANCED\_PACKET\_INJECTION -l FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 –v -r**“ removes (**-r**) the dynamic filter (**-v**) at FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 (**-l**) which references the appropriate callout.

“**WFPSampler.Exe -s ADVANCED\_PACKET\_INJECTION -l FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 -ipla 1.0.0.1 –ipra 1.0.0.254**“ adds a persistent filter at FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 (**-l**) which references the appropriate callout.  This filter will have 2 conditions; FWPM\_CONDITION\_IP\_LOCAL\_ADDRESS (**-ipla**) equals 1.0.0.1, and FWPM\_CONDITION\_IP\_REMOTE\_ADDRESS (**-ipra**) equals 1.0.0.254.

“**WFPSampler.Exe -s ADVANCED\_PACKET\_INJECTION -l FWPM\_LAYER\_INBOUND\_TRANSPORT\_V4 -ipla 1.0.0.1 –ipra 1.0.0.254 –ipp TCP -in**“ adds a persistent filter at FWPM\_LAYER\_INBOUND\_IPPACKET\_V4  (**-l**) which references the appropriate callout.  This filter will have 3 conditions; FWPM\_CONDITION\_IP\_LOCAL\_ADDRESS (**-ipla**) equals 1.0.0.1, FWPM\_CONDITION\_IP\_REMOTE\_ADDRESS (**-ipra**) equals 1.0.0.254, and FWPM\_CONDITION\_IP\_PROTOCOL (**-ipp**) equals TCP.  The injection will be performed synchronously (**-in**).

For a list of conditions applicable to each layer, refer to Filtering Conditions Available at Each Filtering Layer.

For a list of command line parameters for configuring each condition, refer to Conditions for Command Line.