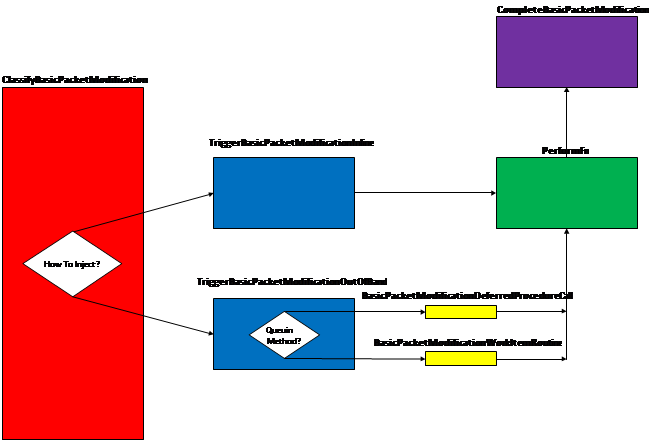
BASIC PACKET MODIFICATION

**Overview**

The Basic Packet Modification scenario will clone the packet and inject it back to the same layer.  Any requested modifications will be performed prior to the injection

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 Basic Packet Modification Scenario**

When traffic matches a filter at the specified layer, **ClassifyBasicPacketModification()** 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), **TriggerBasicPacketModificationInline()** 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), **TriggerBasicPacketModificationOutOfBand()** 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 (default is DPC), the appropriate queueFn 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, the original is cloned, and the offsets of the original are returned to the original place.  Any modification requested will be done to the clone.  Finally, the injectionFn is invoked.

Upon successful injection, **CompleteBasicPacketModification()** 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 | BASIC\_PACKET\_MODIFICATION | Implement the BASIC\_PACKET\_MODIFICATION scenario |
| -l | Applicable Layer | Layer at which this filter will apply |
| -mmsa | MAC Address | Modify the source MAC address to the specified MAC address (Ethernet layers only) |
| -mmda | MAC address | Modify the destination MAC address to the specified MAC address (Ethernet layers only) |
| -misa | IP address | Modify the source IP address to the specified IP address |
| -mida | IP address | Modify the destination IP address to the specified IP address |
| -mtsp | Port | Modify the source port to the specified port (UDP / TCP) or ICMP Type |
| -mtdp | Port | Modify the destination port to the specified port (UDP / TCP) or ICMP Type |
| -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 |
| -? |  | Display help |

“**WFPSampler.Exe -s BASIC\_PACKET\_MODIFICATION -?**“ provides help output

“**WFPSampler.Exe -s BASIC\_PACKET\_MODIFICATION -l FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 -v -mida 1.0.0.1**“ 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.  The destination IP address for all packets will be modified to 1.0.0.1.

“**WFPSampler.Exe -s BASIC\_PACKET\_MODIFICATION -l FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 -v -mida 1.0.0.1 -r**“ removes (**-r**) the dynamic filter (**-v**) at FWPM\_LAYER\_INBOUND\_IPPACKET\_V4 (**-l**) which references the appropriate callout.

“**WFPSampler.Exe -s BASIC\_PACKET\_MODIFICATION -l FWPM\_LAYER\_INBOUND\_IPPACKET\_V4  -ipla 1.0.0.1 -ipra 1.0.0.254 -mida 2.0.0.1**“ 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.  Packets matching this filter will have their destination IP address changed to 2.0.0.1.

“**WFPSampler.Exe -s BASIC\_PACKET\_MODIFICATION -l FWPM\_LAYER\_INBOUND\_TRANSPORT\_V4  -ipla 1.0.0.1 -ipra 1.0.0.254 -ipp TCP -mida 2.0.0.1 -mtdp 35000 -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.  All packets matching this filter will have their destination IP address changed to 2.0.0.1 and their TCP destination port changed to 35000.  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.