FLAIM

LOG ANONYMIZATION

FLAIM Modules
Users Guide

LAIM Group

National Center for Supercomputing Applications

University of Illinois, Urbana-Champaign

February 28, 2008

Contents

Chapter 1

Overview of FLAIM Modules

1.1 The Need for Module User's Guide

FLAIM–Core provides a set of anonymization algorithms for specific data-types as well as policy management for the module developer. However, we (the *LAIM Working Group*) cannot predict the fields that will compose data sources for which third parties will write future modules. A policy specifies the fields which are being anonymized and how they are anonymized. The **FLAIM**–Core schema ensures that a policy uses valid anonymization algorithms with valid options. In addition to this schema, the module developer must write a schema. This module schema ensures that the fields specified in the policy exist in the data source for which the module was written, and that the anonymization algorithms make sense for those types of fields. For example, one would not want to perform regular expression string based substitution on a binary valued field. Only the module developer, who has chosen the field names, knows the data-type for that field and hence what algorithms are applicable. Thus, they control which algorithms are available for which fields in the module schema.

A **FLAIM** user must know what a valid policy looks like, and as shown above, this requires information specific to the module (e.g., the field names and the algorithms applicable to those specific fields. A user could look at the module schema to learn this information, but we provide this information in a simple, tabular form in this guide. Each subsequent chapter covers a specific **FLAIM** module. We also show a general example policy in figure 1 which we will modify for an example in each chapter for that chapter's specific module. This policy below demonstrates the basic format of a **FLAIM** policy and works as our starting point.

1.2 Sample Policy

```
<policy>

  <input>/home/user/input.log</input>
  <output>/home/user/anonymized.log</output>

  <field name="FIELD-1">
    <IPv4PrefixPreserving>
      <Passphrase>disneyland</Passphrase>
    </IPv4PrefixPreserving>
  </field>

  <field name="FIELD-2">
    <NumericTruncation>
      <numShifts>16</numShifts>
      <radix>2</radix>
    </NumericTruncation>
  </field>

  <field name="FIELD-3">
    <RandomPermutation></RandomPermutation>
  </field>

  <field name="FIELD-4">
    <RandomTimeShift>
      <lowerTimeShiftLimit>0</lowerTimeShiftLimit>
      <upperTimeShiftLimit>60</upperTimeShiftLimit>
    </RandomTimeShift>
  </field>

  <field name="FIELD-5">
    <TimeUnitAnnihilation>
      <timeField>seconds</timeField>
    </TimeUnitAnnihilation>
  </field>

  <field name="FIELD-6">
    <TimeEnumeration>
      <bufferSize>25</bufferSize>
      <intervalSize>1</intervalSize>
```

```
40      </TimeEnumeration>
41    </field>
42
43    <field name="FIELD-7">
44      <BlackMarker>
45        <numMarks>7</numMarks>
46        <replacement>0</replacement>
47      </BlackMarker>
48    </field>
49
50    <field name="FIELD-8">
51      <Classify>
52        <configString>9:9,99:99,999:999,1024:1024,9999:9999</configString>
53      </Classify>
54    </field>
55
56    <field name="FIELD-9">
57      <HostBlackMarker>
58        <Type>HostOnly</Type>
59        <hostReplacement>foo</hostReplacement>
60        <domainReplacement>bar</domainReplacement>
61      </HostBlackMarker>
62    </field>
63
64    <field name="FIELD-10">
65      <HostHash>
66        <type>MD5</type>
67      </HostHash>
68    </field>
69
70  </policy>
```

1.2.1 Explanation of Policy File

The sample policy above demonstrates a **FLAIM** policy for some generic log with field names *FIELD-1* through *FIELD-10*, one field to illustrate each of our anonymization algorithms. As we can see, each policy starts and ends with the *policy* tags (line 1 and 70). This indicates the beginning and the end of a **FLAIM** policy. Lines 3 and 4 illustrate two optional tags: *input* and *output*. These tags are used to specify the source and destination logs for **FLAIM** to process. Every other tag is either a *field* tag or subordinate to a *field* tag. These tags are used to specify what algorithms to apply to which fields. Each *field* tag has an attribute named *name* (e.g., lines 6, 12, and 19 in the example policy above). This attribute specifies the name for the field as chosen by the module developer. In this generic example, we use names of the form *FIELD-X* where *X* is an integer.

Understanding the options within a *field* tag requires context. The subsequent chapters of this guide and the **FLAIM**–Core User's Guide give that context. If we look at the example of *FIELD-1* (lines 6–10), we see that we are applying prefix-preserving anonymization for IPv4 addresses (lines 7 and 9). We could look in the **FLAIM**–Core User's Guide (Appendix A) for more information on that algorithm, named *IPv4Prefix-Preserving*. In this example, we see it is passed an option called *Passphrase* (line 8) with a value of "disneyland". Again, the **FLAIM**–Core User's Guide would explain the valid options and their possible values. In this specific case, the passphrase is a seed to a prefix-preserving mapping on IP addresses.

While the **FLAIM**–Core User's Guide is invaluable for explaining the syntax of the various anonymization algorithms, one must use this guide to see what field names are acceptable and which algorithms are available for those fields. If this were not a fictitious sample, there would be a chapter in this guide to explain that there is a field named *FIELD-1*, and that *IPv4Prefix-Preserving* anonymization can be used on that field.

Chapter 2

Pcap Module

Libpcap is a common library use to read and write packets traces to disk. The pcap format has become standard for storing packet traces and is used by *tcpdump*, *ethereal*, *snort* and many other security and networking tools. Pcap traces store all the data related to a network connection and can even be replayed on a network device with utilities like *tcpreplay*

The **FLAIM** pcap module parses pcap traces and passes the packets to **FLAIM**–Core for header anonymization. It does not parse any packet payload, leaving it as a single chunk. Future versions of this module may support "packet cooking" and anonymize application layer data for well-structured and popular protocols (e.g., HTTP and FTP). Most IP (layer 3) header fields can be anonymized, as well as the transport layer header fields for ICMP, TCP and UDP packets.

2.1 Valid Pcap Fields to Anonymize

Table 1 shows the valid field names, their descriptions and the allowable anonymization algorithms for those fields. Unfortunately, we had to print this table in landscape format due to its width, and thus it is almost necessary to print this manual. The descriptions of these algorithms and the parameters for them are described in the **FLAIM**–Core User's Guide. Chapter 1 of this guide demonstrates the general format of a valid policy. Together, these resources allow one to write a valid pcap anonymization policy for **FLAIM** .

Field Name	Short Description	Data Type	Anonymization Algorithm	Comments
`SRC_MAC`	Source MAC address	Byte array	• BinaryBlackMarker • BinaryTruncation • Annihilation • BinaryRandomPermutation • Hash	none
`DST_MAC`	Destination MAC address	Byte array	• BinaryBlackMarker • BinaryTruncation • Annihilation • BinaryRandomPermutation • Hash	none
`IPV4_SRC_IP`	Source IP address	uint32	• BinaryPrefixPreserving • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • NumericTruncation • Hash	none

IPV4_DST_IP	Destination IP address	uint32	• BinaryPrefixPreserving • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • NumericTruncation • Hash	none
IPV4_ID	IPv4 identification	uint16	• BinaryBlackMarker • Annihilation • NumericTruncation • BinaryRandomPermutation • Classify • Hash	none
IPV4_OFFSET	IPv4 fragment offset	uint16	• Annihilation • Hash	none
IPV4_TTL	IPv4 time to live	uint8	• BinaryBlackMarker • Annihilation • NumericTruncation • BinaryRandomPermutation • Classify • Hash	none
IPV4_CHECKSUM	IPv4 checksum	uint16	• Annihilation • Hash	none

`TCP_DST_PORT`	Destination port	uint32	• BinaryBlackMarker • NumericTruncation • Substitution • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`TCP_SRC_PORT`	Source port	uint32	• BinaryBlackMarker • NumericTruncation • Substitution • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`TCP_SEQUENCE`	Sequence number	uint32	• BinaryBlackMarker • NumericTruncation • Annihilation • Classify • Hash	none
`TCP_ACK_NO`	Acknowledgement number	uint32	• BinaryBlackMarker • NumericTruncation • Annihilation • Classify • Hash	none

TCP_FLAGS	Flags	uint8	• BinaryBlackMarker • NumericTruncation • Annihilation • Hash	none
TCP_WINDOW	Window	uint16	• BinaryBlackMarker • NumericTruncation • Annihilation • Classify • Hash	none
TCP_CHECKSUM	Checksum	uint16	• Annihilation • Hash	none
TCP_URGENT	Urgent pointer	uint16	• BinaryBlackMarker • NumericTruncation • Annihilation • Classify • Hash	none
TCP_OPTIONS	Additional header fields	byte array	• Annihilation • Hash	none

`UDP_DST_PORT`	UDP Destination Port	uint16	• BinaryBlackMarker • NumericTruncation • Substitution • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`UDP_SRC_PORT`	UDP Source Port	uint16	• BinaryBlackMarker • NumericTruncation • Substitution • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`UDP_CHECKSUM`	UDP Checksum	uint16	• Annihilation • Hash	none
`ICMP_TYPE`	ICMP Type	uint8	• BinaryBlackMarker • NumericTruncation • Annihilation • BinaryRandomPermutation • Classify • Hash	none

ICMP_CODE	ICMP Code	uint16	• BinaryBlackMarker • NumericTruncation • Annihilation • BinaryRandomPermutation • Classify • Hash	none
ICMP_CHECKSUM	ICMP Checksum	uint16	• Annihilation • Hash	none
ICMP_IDENTIFIER	ICMP Type	uint8	• BinaryBlackMarker • NumericTruncation • Annihilation • BinaryRandomPermutation • Classify • Hash	none
ICMP_SEQUENCE	ICMP Sequence	uint16	• BinaryBlackMarker • NumericTruncation • Annihilation • BinaryRandomPermutation • Classify • Hash	none

`ICMP_GATEWAY`	ICMP Gateway	uint32	• BinaryPrefixPreserving • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • NumericTruncation • Classify • Hash	none
`ICMP_POINTER`	ICMP Pointer	uint8	• BinaryBlackMarker • Annihilation • BinaryRandomPermutation • NumericTruncation • Classify • Hash	none
`ICMP_ORIG_DATA`	ICMP original data	Byte array	• BinaryBlackMarker • Annihilation • Hash	none
`ICMP_TS_ORIG`	ICMP originate timestamp	uint32	• RandomTimeShift • TimeUnitAnnihilation • Annihilation • BinaryBlackMarker • TimeEnumeration • Hash	none

`ICMP_TS_REC`	ICMP receive timestamp	uint32	• RandomTimeShift • TimeUnitAnnihilation • Annihilation • BinaryBlackMarker • TimeEnumeration • Hash	none
`ICMP_TS_TRANS`	ICMP transmit timestamp	uint32	• RandomTimeShift • TimeUnitAnnihilation • Annihilation • BinaryBlackMarker • TimeEnumeration • Hash	none
`ICMP_IPV4_SRC_IP`	ICMP IPv4 source ip	uint32	• BinaryPrefixPreserving • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • NumericTruncation • Hash	none
`ICMP_IPV4_DST_IP`	ICMP IPv4 destination ip	uint32	• BinaryPrefixPreserving • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • NumericTruncation • Hash	none

`ICMP_IPV4_ID`	ICMP IPv4 identification	uint16	• BinaryBlackMarker • Annihilation • NumericTruncation • BinaryRandomPermutation • Classify • Hash	none
`ICMP_IPV4_OFFSET`	ICMP IPv4 fragment offset	uint16	• Annihilation • Hash	none
`ICMP_IPV4_TTL`	ICMP IPv4 time to live	uint8	• BinaryBlackMarker • Annihilation • NumericTruncation • BinaryRandomPermutation • Classify • Hash	none
`ICMP_IPV4_CHECKSUM`	ICMP IPv4 checksum	uint16	• Annihilation • Hash	none
`TS_SEC`	Packet capture time, in seconds since epoch	uint32	• RandomTimeShift • TimeUnitAnnihilation • Annihilation • BinaryBlackMarker • TimeEnumeration • Hash	none

TS_USEC	Microsecond of packet capture time - offset of TS_SEC	uint32	• Annihilation • Hash	none

2.2 Example Policy

Below we show a simple policy that anonymizes the timestamps by shifting them, IP addresses by prefix-preserving pseudonymization and removes any TCP options.

```
<policy>

  <field name="TS_SEC">
    <RandomTimeShift>
      <lowerTimeShiftLimit>0</lowerTimeShiftLimit>
      <upperTimeShiftLimit>31000000</upperTimeShiftLimit>
    </RandomTimeShift>
  </field>

  <field name="IPV4_SRC_IP">
    <IPv4Prefix-Preserving>
      <Passphrase>Passw0rd</Passphrase>
    </IPv4Prefix-Preserving>
  </field>

  <field name="IPV4_DST_IP">
    <IPv4Prefix-Preserving>
      <Passphrase>Passw0rd</Passphrase>
    </IPv4Prefix-Preserving>
  </field>

  <field name="TCP_OPTIONS">
    <Annihilation></Annihilation>
  </field>

</policy>
```

Chapter 3

Netfilter/iptables Module

Netfilter (often called iptables) is a kernel level NAT and firewall implementation for Linux 2.4 and 2.6 kernels. For any rule that is matched, regardless of the action taken, the packets that match can be logged via syslog. **FLAIM** supports anonymization of these syslog messages created by netfilter.

FLAIM can anonymize the layer 2 and 3 information recorded via netfilter as well as layer 4 header data for TCP, UDP and ICMP protocols. There are just a couple of layer 4 protocols parsed by netfilter that **FLAIM** will ignore (e.g., protocols 50 and 51 for IPSEC).

Unlike some of the modules for binary log formats, this module will accept streamed data. If no input or output file is specified on the **FLAIM** command line or in the XML policy, this module will read input from STDIN and write to STDOUT. This makes it simpler to script anonymization of records or to write them to disk anonymized.

3.1 Valid Netfilter/Iptables Fields to Anonymize

Table 2 shows the valid field names, their descriptions and the allowable anonymization algorithms for those fields. Unfortunately, we had to print this table in landscape format due to its width, and thus it is almost necessary to print this manual. The descriptions of these algorithms and the parameters for them are described in the **FLAIM**–Core User's Guide. Chapter 1 of this guide demonstrates the general format of a valid policy. Together, these resources allow one to write a valid netfilter/iptables anonymization policy for **FLAIM** .

Field Name	Short Description	Data Type	Anonymization Algorithm	Comments
SYS_TRANS_TYPE	Protocol type	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none
PCKT_TS_SEC	Timestamp	uint32	• RandomTimeShift • TimeUnitAnnihilation • TimeEnumeration • BinaryBlackMarker • Annihilation • Hash	none
PCKT_MACHINE_NAME	Host running netfilter	string	• HostBlackMarker • HostHash • Annihilation • Substitution • Hash	none
PCKT_LOG_INTERFACE	Log prefix specified to netfilter on the command line	String	• StringBlackMarker • Annihilation • Hash	none

PCKT_IN_INTERFACE	Incoming network interface	string	• StringBlackMarker • Annihilation • Hash	none
PCKT_OUT_INTERFACE	Outgoing network interface	string	• StringBlackMarker • Annihilation • Hash	none
ETHER_SRC_MAC	Source MAC address	byte array	• BinaryTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none
ETHER_DST_MAC	Destination MAC address	byte array	• BinaryTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none
IPV4_SRC_IP	Source IP address	uint32	• BinaryPrefixPreserving • NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none

`IPV4_DST_IP`	Destination IP address	uint32	• BinaryPrefixPreserving • NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none
`IPV4_TOS`	IPv4 type of service	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`IPV4_PRECEDENCE`	Type of service, precedence field (1 byte)	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none

IPV4_TTL	ICMP IPv4 time to live	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
IPV4_ID	IPv4 identification	uint16	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
IPV4_CE	congestion experienced flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none
IPV4_DF	Don't fragment bit	uint8	• BinaryBlackMarker • Annihilation • Hash	none
IPV4_MF	More fragment bit	uint8	• BinaryBlackMarker • Annihilation • Hash	none

IPV4_FRAG	Fragment offset	uint16	• Annihilation • Hash	none
IPV4_OPT	IP-related options	byte array	• BinaryBlackMarker • Annihilation • Hash	none
TCP_SRC_PORT	Source port	uint16	• Substitution • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
TCP_DST_PORT	Destination port	uint16	• Substitution • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none
TCP_SEQUENCE	Sequence number	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • Classify • Hash	none

`TCP_ACK_NO`	Acknowledgement number	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • Classify • Hash	none
`TCP_WINDOW`	Window	uint16	• NumericTruncation • BinaryBlackMarker • Annihilation • Classify • Hash	none
`TCP_FLAG_URGENT`	Urgent pointer field significant	uint8	• BinaryBlackMarker • Annihilation • Hash	none
`TCP_FLAG_ACK`	Acknowledgement field significant	uint8	• BinaryBlackMarker • Annihilation • Hash	none
`TCP_FLAG_PSH`	PSH flag - Push function	uint8	• BinaryBlackMarker • Annihilation • Hash	none
`TCP_FLAG_RST`	RST flag - Reset the connection	uint8	• BinaryBlackMarker • Annihilation • Hash	none

TCP_FLAG_SYN	SYN flag - Synchronize sequence numbers	uint8	• BinaryBlackMarker • Annihilation • Hash	none
TCP_FLAG_FIN	FIN flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none
TCP_URGENT_PTR	Urgent pointer	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • Classify • Hash	none
TCP_OPTIONS	Additional header fields	byte array	• BinaryBlackMarker • Annihilation • Hash	none
UDP_SRC_PORT	UDP Source Port	uint16	• BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none

UDP_DST_PORT	UDP Destination Port	uint16	• BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none
ICMP_TYPE	ICMP Type	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
ICMP_CODE	ICMP Code	uint16	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none

`ICMP_EXT_ID`	ICMP ID number	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`ICMP_EXT_SEQ`	ICMP sequence number	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • Classify • Hash	none
`ICMP_SRC_IP`	ICMP Source IP address.	uint16	• BinaryPrefixPreserving • NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none
`ICMP_DST_IP`	ICMP IPv4 destination ip	uint32	• BinaryPrefixPreserving • NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none

ICMP_TOS	ICMP Type of service flags	uint8	• BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
ICMP_PRECEDENCE	Encapsulated precedence data	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
ICMP_TTL	ICMP IPv4 time to live	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none

ICMP_ID	ICMP IPv4 identification	uint16	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
ICMP_CE	Encapsulated congestion flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none
ICMP_DF	Don't fragment bit	uint8	• BinaryBlackMarker • Annihilation • Hash	none
ICMP_MF	More fragment bit	uint8	• BinaryBlackMarker • Annihilation • Hash	none
ICMP_FRAG	Fragment offset	uint16	• Annihilation • Hash	none
ICMP_OPT	Options field	byte array	• BinaryBlackMarker • Annihilation • Hash	none

`ICMP_TCP_SRC_PORT`	Source port	uint16	• BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none
`ICMP_TCP_DST_PORT`	Destination port	uint16	• BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none
`ICMP_TCP_SEQUENCE`	Sequence number	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • Classify • Hash	none
`ICMP_TCP_ACK_NO`	Acknowledgement number	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • Classify • Hash	none

ICMP_TCP_WINDOW	Window	uint16	• NumericTruncation • BinaryBlackMarker • Annihilation • Classify • Hash	none
ICMP_TCP_FLAG_URGENT	Urgent flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none
ICMP_TCP_FLAG_ACK	Acknowledgement flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none
ICMP_TCP_FLAG_PSH	Psh flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none
ICMP_TCP_FLAG_RST	Reset flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none
ICMP_TCP_FLAG_SYN	SYN flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none
ICMP_TCP_FLAG_FIN	FIN flag	uint8	• BinaryBlackMarker • Annihilation • Hash	none

`ICMP_TCP_URGENT_PTR`	Encapsulated pointer to urgent data	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`ICMP_TCP_OPTIONS`	TCP options	uint32	• BinaryBlackMarker • Annihilation • Hash	none
`ICMP_UDP_SRC_PORT`	UDP source port	uint16	• BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none
`ICMP_UDP_DST_PORT`	UDP destination port	uint16	• BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none

`ICMP_TYPE_INTERNAL`	Encapsulated datagram type	ICMP	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`ICMP_CODE_INTERNAL`	Encapsulated message code	ICMP	uint8	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none
`ICMP_ID_INTERNAL`	Encapsulated ID number	ICMP	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none

ICMP_SEQ_INTERNAL	Encapsulated ICMP sequence number	uint32	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Hash	none

3.2 Example Policy

Below we show a simple policy that anonymizes the timestamps by the enumeration method, IP addresses by truncating the last octet and blacks out the MAC addresses.

```
<policy>

  <field name="PCKT_TS_SEC">
    <TimeEnumeration>
      <intervalSize>1</intervalSize>
      <bufferSize>100</bufferSize>
    </TimeEnumeration>
  </field>

  <field name="IPV4_SRC_IP">
    <NumericTruncation>
      <numShifts>16</numShifts>
      <radix>2</radix>
    </NumericTruncation>
  </field>

  <field name="IPV4_DST_IP">
    <NumericTruncation>
      <numShifts>16</numShifts>
      <radix>2</radix>
    </NumericTruncation>
  </field>

  <field name="ETHER_SRC_MAC">
    <BlackMarker>
      <type>byte</type>
      <numMarks>6</numMarks>
      <replacement>0</replacement>
    </BlackMarker>
  </field>

  <field name="ETHER_DST_MAC">
    <BlackMarker>
      <type>byte</type>
      <numMarks>6</numMarks>
      <replacement>0</replacement>
```

```
    </BlackMarker>
  </field>

</policy>
```

Chapter 4

Nfdump Module

A NetFlow is simply a way of abstracting network traffic to the level of a flow rather than individual packets. Usually, there is a one-to-one correspondence between NetFlows and sockets. So a NetFlow is uniquely identified by source and destination IP addresses and ports, though all the packets that comprise a particular flow must traverse the router within a certain time window. So a socket that happens to involve the same ports and IP addresses, but is opened on a different day, will create a separate flow record. Several fields are common to all NetFlow formats: source IP, destination IP, source port, destination port, starting timestamp, ending timestamp, bytes transferred and number of packets exchanged.

The nfdump software suite has several tools used to collect and analyze Cisco NetFlows. Nfcapd is a daemon process that collects the flows and writes them into the nfdump format. The nfdump utility works much like tcpdump to let you filter and analyze specific records in an ASCII format. There are several other small tools that come with nfdump to help you manage the flow records nfcapd collects.

There have been changes in the internal nfdump format between minor version numbers, unfortunately. So older versions of this module work with nfdump 1.4.x flows, but not 1.5.x records. To anonymize nfdump version 1.5.x logs, one must use at least FLAIM version 0.6.0.

4.1 Valid Nfdump Fields to Anonymize

Table 3 shows the valid field names, their descriptions and the allowable anonymization algorithms for those fields. Unfortunately, we had to print this table in landscape format due to its width, and thus it is almost necessary to print this manual. The descriptions of these algorithms and the parameters for them are described in the **FLAIM**–Core User's Guide. Chapter 1 of this guide demonstrates the general format of a valid policy. Together, these resources allow one to write a valid nfdump anonymization policy for **FLAIM** .

Field Name	Short Description	Data Type	Anonymization Algorithm	Comments
SRC_IP	Source IP address	uint32	• NumericTruncation • BinaryPrefixPreserving • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none
DST_IP	Destination IP address	uint32	• NumericTruncation • BinaryPrefixPreserving • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none
NEXT_HOP	Next hop router	uint32	• NumericTruncation • BinaryPrefixPreserving • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Hash	none
INPUT	SNMP index of input interface	uint16	• BinaryBlackMarker • Annihilation • Hash	none

`OUTPUT`	SNMP index of output interface	uint16	• BinaryBlackMarker • Annihilation • Hash	none
`PACKETS`	Number of Packets in flow	uint32	• BinaryBlackMarker • Annihilation • Hash	none
`OCTETS`	Number of layer 3 bytes in the packets of the flow	uint32	• BinaryBlackMarker • Annihilation • Hash	none
`TS_SEC_FIRST`	System uptime at start of flow, in seconds	uint32	• RandomTimeShift • TimeEnumeration • TimeUnitAnnihilation • Annihilation • Hash	none
`TS_SEC_LAST`	System uptime at time where last packet was received, in seconds	uint32	• RandomTimeShift • TimeEnumeration • TimeUnitAnnihilation • Annihilation • Hash	none
`HEAD_SYS_UPTIME`	Current time in milliseconds since the export device booted	uint32	• RandomTimeShift • TimeEnumeration • TimeUnitAnnihilation • Annihilation • Hash	none

HEAD_UNIX_SECS	Current count of seconds since 0000 UTC 1970	uint32	• RandomTimeShift • TimeEnumeration • TimeUnitAnnihilation • Annihilation • Hash	none
HEAD_UNIX_NSECS	Residual nanoseconds since 0000 UTC 1970	uint32	• RandomTimeShift • TimeEnumeration • TimeUnitAnnihilation • Annihilation • Hash	none
SRC_PORT	Source port	uint16	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none
DST_PORT	Destination port	uint16	• NumericTruncation • BinaryBlackMarker • Annihilation • BinaryRandomPermutation • Classify • Substitution • Hash	none

`TCP_FLAGS`	TCP flags	uint8	• BinaryBlackMarker • Annihilation • Hash	If processing v5 flows, this will be a cumulative OR of the TCP flags. In v7 flows, it is always set to 0
`FLAGS1`	Flags	uint8	• BinaryBlackMarker • Annihilation • Hash	First flag field in byte 36 of v7 flows only.
`FLAGS2`	Flags	uint16	• BinaryBlackMarker • Annihilation • Hash	Second flag field at bytes 46-47 of v7 flows only
`PROTOCOL`	IP Protocol.	uint8	• BinaryBlackMarker • Annihilation • Classify • BinaryRandomPermutation • Hash	for v7 flows: set to zero if flow mask is destination-only or source-destination

4.2 Example Policy

Below we show a simple policy that anonymizes the timestamps annihilating the second information, IP addresses by random permutation and removes the number of bytes transferred.

```
<policy>

  <field name="TS_SEC_FIRST">
    <TimeUnitAnnhilation>
      <timeField>seconds</timeField>
      <secondaryField>TS_SEC_LAST</secondaryField>
    </TimeUnitAnnhilation>
  </field>

  <field name="IPV4_SRC_IP">
    <RandomPermutation></RandomPermutation>
  </field>

  <field name="IPV4_DST_IP">
    <RandomPermutation></RandomPermutation>
  </field>

  <field name="OCTETS">
    <Annihilation></Annihilation>
  </field>

</policy>
```

4.3 Copyrights and Acknowledgments

This module in particular uses source code developed by SWITCH, and therefore we present the copyright notice below.

Chapter 5

Pacct Module

Process accounting events can be recorded on most UNIX-like operating systems, though such recording must usually be turned-on manually. Often this requires kernel recompilation as well. Process accounting events are generated one per process and record things such as the command executed, the user and group owning the process, the time spent in both user and kernel space, and the time and date when the process was created. On all supporting platforms, one can use the *lastcomm* utility to access these records. On Linux and many BSDs, one can use the *acct* utilities suite for more thorough analysis of process accounting logs. This GNU suite of tools includes *dump-acct*—a tool to read and convert RAW records—and other useful tools that do far more than *lastcomm*.

Because each OS implements process accounting differently, it is problematic to create a tool that handles all the different varieties. On Linux alone there are 4 versions with a fifth in the works. Support for many of the BSD's could be added, but the module would have to be compiled on a system with the kernel headers in place so that it could read the length of certain fields—like the maximum number of characters to record of the command name and arguments. However, once compiled, it would only work for records of that platform. So compiling FLAIM's module on FreeBSD would mean that it would expect records of a particular length and it may not work on processing logs from an OpenBSD machine. Because of this and other issues, we have made this a Linux process accounting module. It will work on any platform that Linux runs upon, but it will only work with Linux process accounting logs. So even if run on a Mac, it would expect Linux records. Therefore, it is appropriate to call this a *Linux process accounting* module.

As mentioned, there are several Linux process accounting formats. They are described below:

- **v0** — This is the original Linux format and very much mirrors the original BSD style in its kernel structure definition. It is 64 bytes long, but has some extra padding at the end. It is supported by FLAIM.
- **v1** — This is a peculiar format for Mac68K Linux only, that is for the pre-PowerPC

Macintosh computers running Linux. It is incomplete and essentially unused. We do not support it.

- **v2** — This is an extension of the v0 format that uses the extra padded bytes to store extra information (e.g., version, 16 MSB's of a 32 bit UID/GID, and some extra timestamp precision). Because of the clever way it is done, it is still 64 bytes long, and utilities that are only aware of v0 logs can still process v2 logs while only losing the extra precision and bits of the larger UID/GIDs. FLAIM supports this format.
- **v3** — This is just becoming the default format in many Linux distributions. Often it still needs to be compiled in with a special kernel option. This format is not backwards compatible, and it has extra information about things such as the parent process. FLAIM supports this version as well.

5.1 Valid pacct Fields to Anonymize

Table 1 shows the valid field names, their descriptions and the allowable anonymization algorithms for those fields. Unfortunately, we had to print this table in landscape format due to its width, and thus it is almost necessary to print this manual. The descriptions of these algorithms and the parameters for them are described in the **FLAIM**–Core User's Guide. Chapter 1 of this guide demonstrates the general format of a valid policy. Together, these resources allow one to write a valid pcap anonymization policy for **FLAIM** .

Field Name	Short Description	Data Type	Anonymization Algorithm	Comments
`AC_FLAG`	Accounting flags	Byte	• Annihilation • Hash	none
`AC_TTY`	Control Terminal	uint16	• Annihilation • Classify • Substitution • BinaryBlackMarker • NumericTruncation • Hash	none
`AC_UID`	Real User ID	uint32	• Annihilation • Classify • Substitution • BinaryBlackMarker • NumericTruncation • Hash	none
`AC_GID`	Real Group ID	uint32	• Annihilation • Classify • Substitution • BinaryBlackMarker • NumericTruncation • Hash	none

AC_BTIME	Process Creation Time	uint32	• RandomTimeShift • TimeUnitAnnihilation • TimeEnumeration • Annihilation • BinaryBlackMarker • Hash	none
AC_UTIME	User Time, in clock ticks	uint16	• Annihilation • Hash	none
AC_STIME	System Time, in clock ticks	uint16	• Annihilation • Hash	none
AC_ETIME	Elapsed Time, in clock ticks	uint32	• Annihilation • Hash	none
AC_MEM	Average Memory Usage ub clicks	uint16	• Annihilation • Hash	none
AC_IO	Chars Transferred by read/write	uint16	• Annihilation • Classify • Substitution • BinaryBlackMarker • NumericTruncation • Hash	none

AC_RW	Blocks Read or Written	uint16	• Annihilation • Classify • Substitution • BinaryBlackMarker • NumericTruncation • Hash	none
AC_MINFLT	Minor Pagefaults	uint16	• Annihilation • Hash	none
AC_MAJFLT	Major Pagefaults	uint16	• Annihilation • Hash	none
AC_SWAPS	Number of Swaps	uint16	• Annihilation • Hash	none
AC_EXITCODE	Exitcode	uint32	• Annihilation • Classify • Substitution • BinaryBlackMarker • NumericTruncation • Hash	none
AC_COMM	Command Name	byte array	• StringTruncation • StringBlackMarker • Hash	none
AC_PID	Process ID	uint32	• Annihilation • Hash	none

`AC_PPID`	Parent Process ID	uint32	• Annihilation • Hash	none

5.2 Example Policy

Below we show a simple policy that anonymizes the beginning timestamps by annihilating the hour information, the command name by truncating off the last 4 characters, and the UID of the process owner by replacing it with 0.

```
<policy>
<field name="AC_COMM">

<StringTruncation>
      <numChars>4</numChars>
      <direction>right</direction>
</StringTruncation>
</field>

 <field name="AC_UID">
     <Annihilation/>
</field>

<field name="AC_BTIME">
   <TimeUnitAnnihilation>
      <timeField>hours</timeField>
      <secondaryField>NONE</secondaryField>
    </TimeUnitAnnihilation>
  </field>

</policy>
```

5.3 Copyrights and Acknowledgements

This specific module,the pacct module, had to be released under the LGPL (GNU Library General Public License) because it uses source code from the GNU Accounting Utilities[1] as well as data structures from the Linux 2.6 kernel source. This is different than the license for FLAIM Core.

[1] `http://www.gnu.org/software/acct/`

[This is the first released version of the library GPL. It is numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

1. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

   A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

   The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

   "Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

2. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

3. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) The modified work must itself be a software library.

   (b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

   (c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

   (d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

   (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

4. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

   Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

   This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

5. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

   If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

6. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

7. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

   You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

   (a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

   (b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
   (c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
   (d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

   For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

8. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:
   (a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
   (b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

9. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its deriva-

tive works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

11. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

12. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

    If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

    It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

    This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

13. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

14. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

15. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

    NO WARRANTY

16. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

17. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Chapter 6

Copyright

6.1 Copyright

Unless otherwise stated in the specific chapter for a module, it is released with the following licensing and terms.