ripfix: Rapid Prototyping and Debugging of IPFIX Applications in Ruby

Brian Trammell <trammell@tik.ee.ethz.ch>
Communication Systems Group, ETH Zürich
The Problem

- IPFIX is based on a machine-efficient but not particularly human-manipulable or easily debuggable representation
  - A AoBp0wzRUMAAAAAADd1QACAkE0gAOAUlBAAIAAQADAAEEAcAAgALAAkBmQAEEAACK7oGSAEAAIrugZP/wAAiu6Bnv//AACK7oGW/8AAIrugZf//wAAiu6BlP//AACK7oGV//8AAIrugZj//wAAiu4Q4QALAUlBAAlIAAQADAAEEAcAAgALAAkBmQAEEAACK7oGcAAIAAIrZIAAQAAiu6Bnf//AACK7oGW//8AAIrugZf//wAAiu4E0gCms7PydMCoAAPAqAACl...

- IPFIX implementations often optimized for performance, not “tweakability”

- Lots of things that would lead to greater implementation, application, and adoption of IPFIX are harder than they should be:
  - Testing, debugging, disambiguating interoperability issues
  - Trying out new ideas, extensions, IEs, applications, etc…

Wednesday, July 28, 2010
The solution

- Need an IPFIX implementation optimized for development, as opposed to production use:
  - Ease of building new applications that use IPFIX → general “ease of (developer) use”
  - Ease of building new general IPFIX tools for debugging → applications with no internal data model
  - Ease of modifying the implementation itself → rapid prototyping of protocol extensions
- Support “rough consensus and (rapid) running code.”
Ruby

- Duck-typed, object-oriented, interpreted* language
- Suited to rapid development, not rapid processing
- Relatively complete standard library
  - Socket interface easy to use → good for little network apps
- Simple to write methods/classes in C
  - No SCTP support, no problem → ripfix provides SCTP::Endpoint
Design

- Principle of Least Surprise
- Classes map to entities in the protocol
  - Message, Template, OptionsTemplate, InfoElement, etc.
- Data records written/read as Ruby Hashes of Symbols to native Ruby types
  - String, Fixnum, IPAddr, etc.
- Use of Ruby Enumerable idiom (each())
- Templates and Information Models specified as strings
  - defined in draft-trammell-ipfix-text-iespec-00
  - also supports IANA XML registry, but this is somewhat slow
- Type system for transcoding
  - Information Element has a type, which can encode/decode value
Design Details

message.rb

Model.rb

Type.rb

- length
- export_time
- sequence
- domain
- record_count

Operation

- Operation

SetBuffer

- set_id
- accept_id?

Buffer

- length
- a
- append_avail
- shift
- shift_avail
- rewind
- limit
- limit=
- string
- dump

Template

- tid
- count
- min_length
- @elements
- <<
- each
- spec
- encode_template_record
- decode_template_record
- encode_hash
- decode_hash

Options Template

- scope_count
- add_scope

InfoElement

- name
- pen
- number
- size
- hashkey
- spec
- parse_spec
- for_size
- for_reverse
- is_ep?
- is_varlen?

InfoModel

- Types
- TypesByName
- load
- load_v9_compatible
- load_default
- load_reverse
- load_xml
- save
- type_for_number
- type_for_name
- ie_for_number
- ie_for_name
- ie_for_spec
- add
- each

Types

- append_avail
- shift
- shift_avail
- rewind
- limit
- string
- dump

InfoModel

- Type
- name
- number
- size
- encode
- decode
- parse

Type

- UnsignedType
- SignedType
- BooleanType
- FloatType
- IPAddressType
- MACAddressType
- SecondsType
- MillisecondsType
- NTPType
Supports

- RFC 5101 (mostly, see next)
- TCP as transport
- RFC 5103 biflows
- RFC 5655 files (basic)
- draft-trammell-ipfix-text-iespec
- going and getting a cup of coffee while you wait for large data sets to process.
Future plans

- Complete IPFIX over SCTP support
- TLS/DTLS (need bindings)
- Template withdrawals
- Faster buffer and type implementation (in C)
- Metadata: 5610, 5655, ipfix-anon
- draft-ietf-ipfix-structured-data
- draft-ietf-ipfix-per-sctp-stream, better stream management
- More rigorous unit testing, resultant bugfixes
- UDP, template expiration and retransmission (messy)
- …
Experiences

- On-the-fly interop testing
  - rfdump tool used to track the at-fault implementation in an interop disagreement without having to drop to a hex editor.

- Slightly less boring demonstrations of IPFIX applications
  - Need to show someone what IPFIX looks like without walking them through a hexdump? rfcollect (rfdump over TCP).

- Generation of examples for SIPCLF
  - Initial requirement: Hacking up IPFIX stuff in WG meetings

- Quick and dirty analysis tools
  - select-and-project, aggregation for visualization, etc.
How to get it

- [http://ripfix.rubyforge.net](http://ripfix.rubyforge.net)
- Under *active* development
  - Interfaces may change at any time for no reason at all
    - but since the design parallels the protocol, not in nonsensical ways
    - Watch out for any class not listed on Slide 6.
  - SCTP support included, but not yet integrated
  - We try to keep broken code out of the released gem, but…
- Please download, suggest, contribute!
  - trammell@tik.ee.ethz.ch
ADDITIONAL SLIDES

SCTP support for Ruby
SCTP::Endpoint

- Attempt to put a Ruby interface atop all features of SCTP.
  - An Endpoint is a 1-to-1 or 1-to-many socket
  - Sends and receives SCTP::Message objects
    - Encapsulates content, stream, remote endpoint
  - SCTP::Socket wraps Endpoint for a more Ruby Sockets-like interface
    - Direct extension would not be particularly clean.
- Interface even more experimental than ripfix (*will* change) and essentially undocumented.
  - Defined for multihoming, but not yet supported
- No interop testing yet.
- Requires (of course) working SCTP on host system.
  - Only tested on Mac OS X, some Linuxes.