SNMP Traffic Measurements

Jürgen Schönwälder

j.schoenwaelder@iu-bremen.de

International University Bremen

Campus Ring 1

28725 Bremen, Germany

http://www.ibr.cs.tu-bs.de/projects/nmrg/

Outline of the Talk

- Motivation of SNMP Measurements
- Background: Characterization of MIB Modules
- Measurement Approach
- Tool Support (`libanon` & `snmpdump`)
- First Results
- Conclusions

1. Motivation

We know SNMP...

- The Simple Network Management Protocol (SNMP) is widely deployed to
  - monitor devices (collect statistics, event reports),
  - control devices (turning knobs), and
  - (to a lesser extent) configure devices
- SNMP technology is well documented and understood (if you care to study the right documents)
- SNMP supports “fancy” features to allow applications to do the right thing (e.g., discontinuity indicators, row creation modes)
- Especially us (the SNMP geeks) know about these nice features . . .

... but not how it is used

- It remains unclear how SNMP is used in practice in **operational** networks
- In particular, we do not know
  - what typical SNMP usage patterns are
  - which features of SNMP are used/not used
  - which MIB objects and data models (MIB modules) are frequently used
  - whether the work done in the IETF and elsewhere is related to the actual usage of this technology

Why is this important?

- Researchers write papers how to improve SNMP or how other technologies (e.g., Web Services) compare to SNMP without having a justified model
- The IETF works on protocol extensions (currently session-based security in ISMS) without knowing network management traffic models (and in the context of ISMS to what extend a session-based approach to security is viable)
- The IETF requires features during MIB design/review without knowing whether they are used in practice

Questions to answer...

- Basic statistics (version used, typical manager/agent ratios, operations used, . . .)
- Relationship between periodic (regular polling) and aperiodic traffic patterns
- Message size and latency distributions
- Concurrency levels (managers performing similar operations on multiple agents concurrently)
- Table retrieval approaches
  - column-by-column vs. row-by-row
  - usage of `getbulk` and its parameters,
  - suppression of index columns,
  - holes (do they exist?) and how are they dealt with
  - . . .

Questions to answer... (cont.)

- Trap-directed polling - myths or reality?
- Identification of popular MIB definitions
- Usage of deprecated and obsolete objects
- Encoding length distributions of various data types
- Counters and discontinuities
- Synchronization and spin locks
- Row creation (dribble-mode vs. one-shot mode)
- Implementation and configuration errors
- . . .

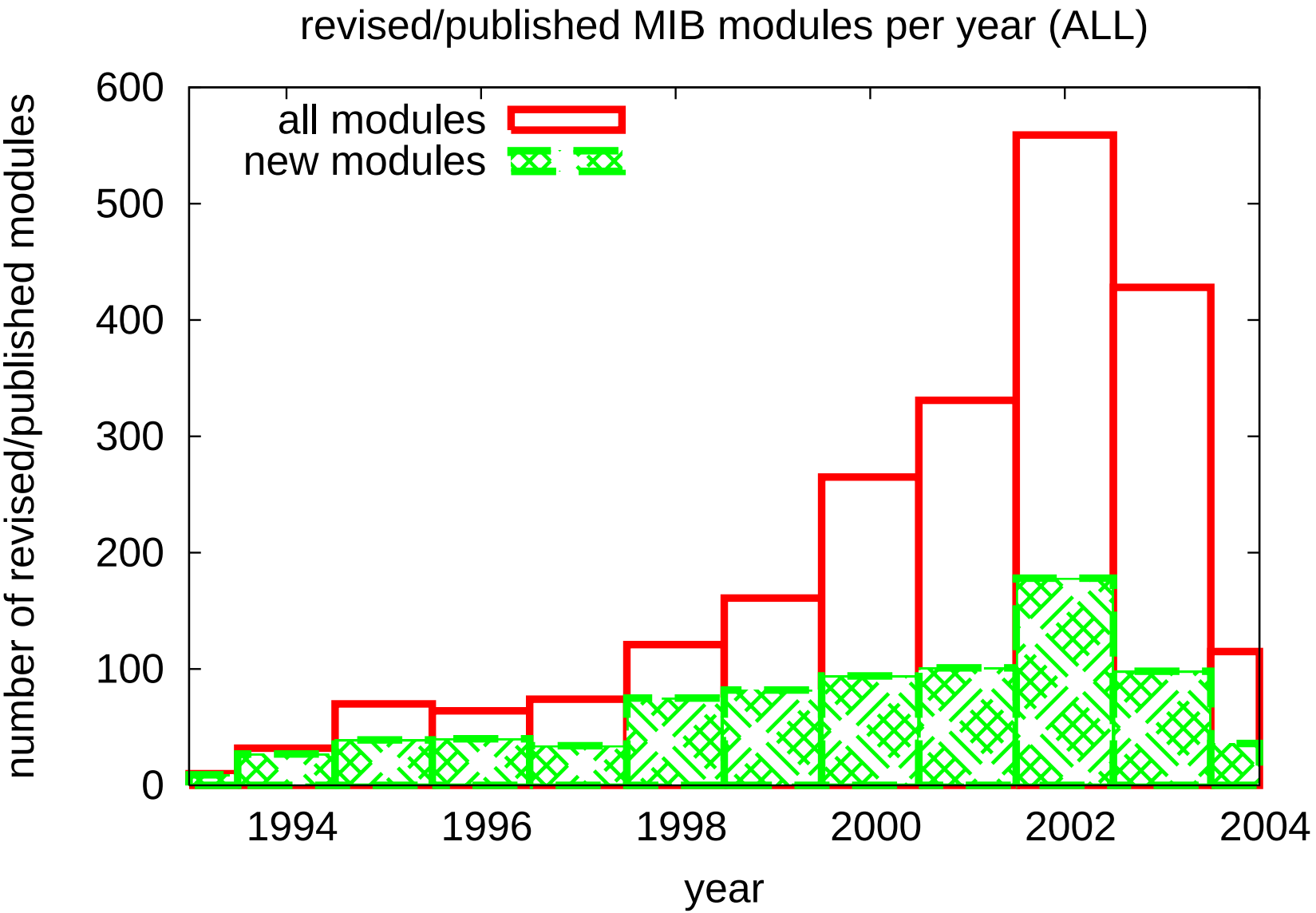
2. Background: Characterization of MIB Modules

Static Analysis of MIB Modules

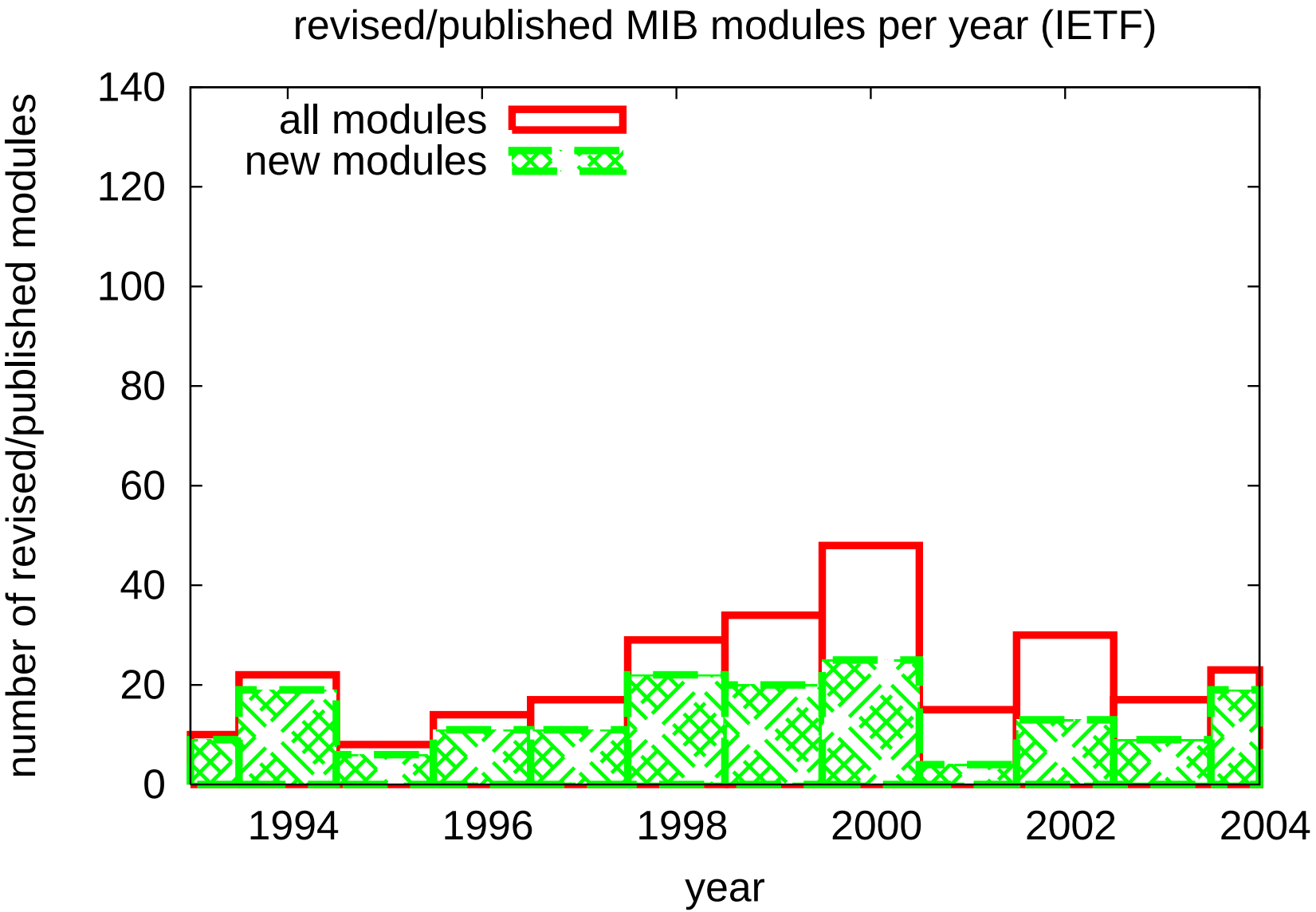
MIB Module Set	Modules	Types	Tables	Columns	Scalars	Notifications
IETF	174	377	875	7479	785	195
ATM Forum	11	63	79	777	39	5
Cisco Systems	482	936	1966	16952	3719	611
Enterasys	58	76	128	825	364	28
Juniper Networks	99	170	434	3606	1051	87
All Modules	824	1622	3482	29639	5958	926

- Characterization of MIB modules performed during summer 2004
- Developed a back-end for the `smidump` MIB compiler to produce various metrics
- Results published at IFIP/IEEE IM 2005 (Nice) [1]

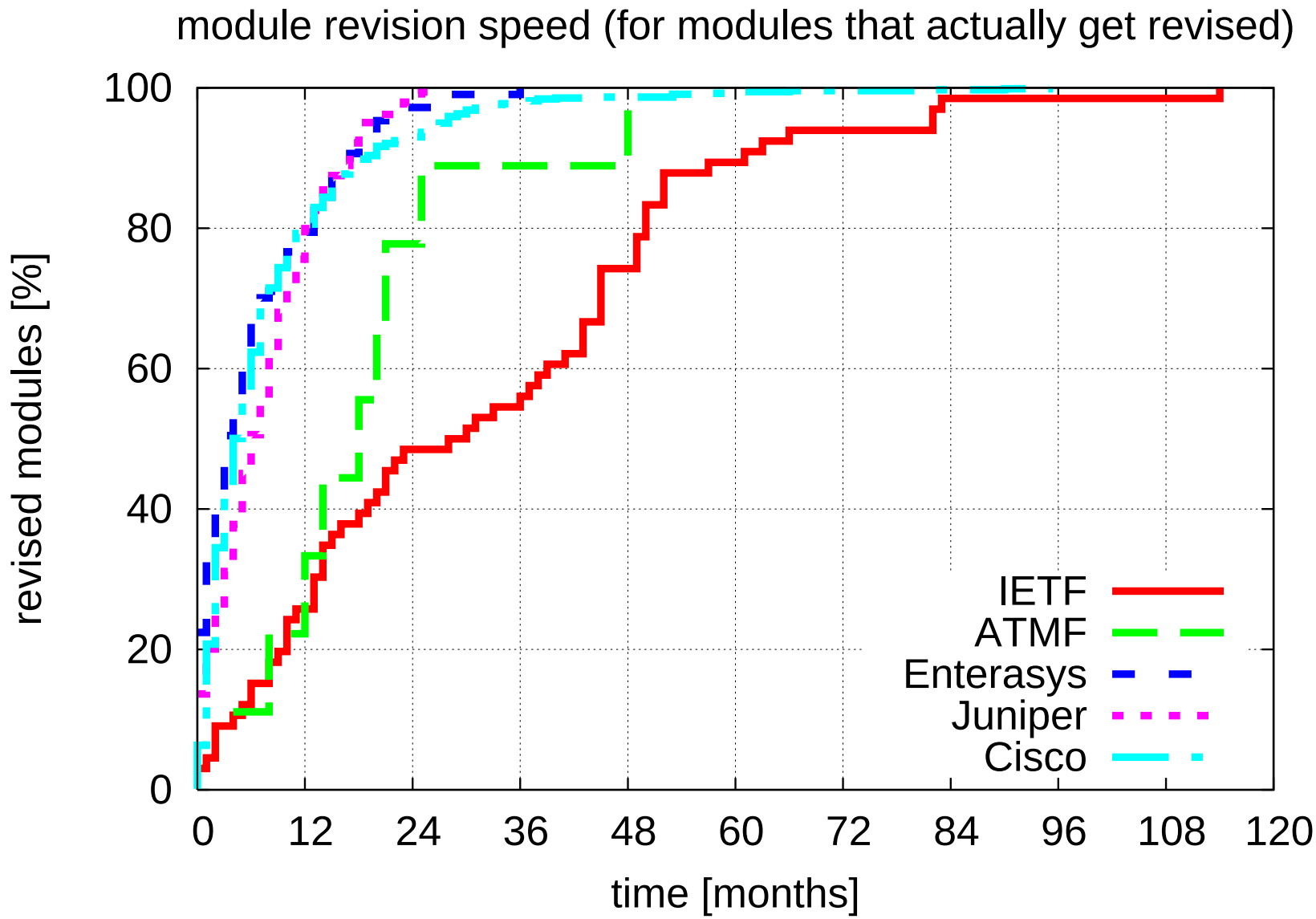
MIB Module Productivity



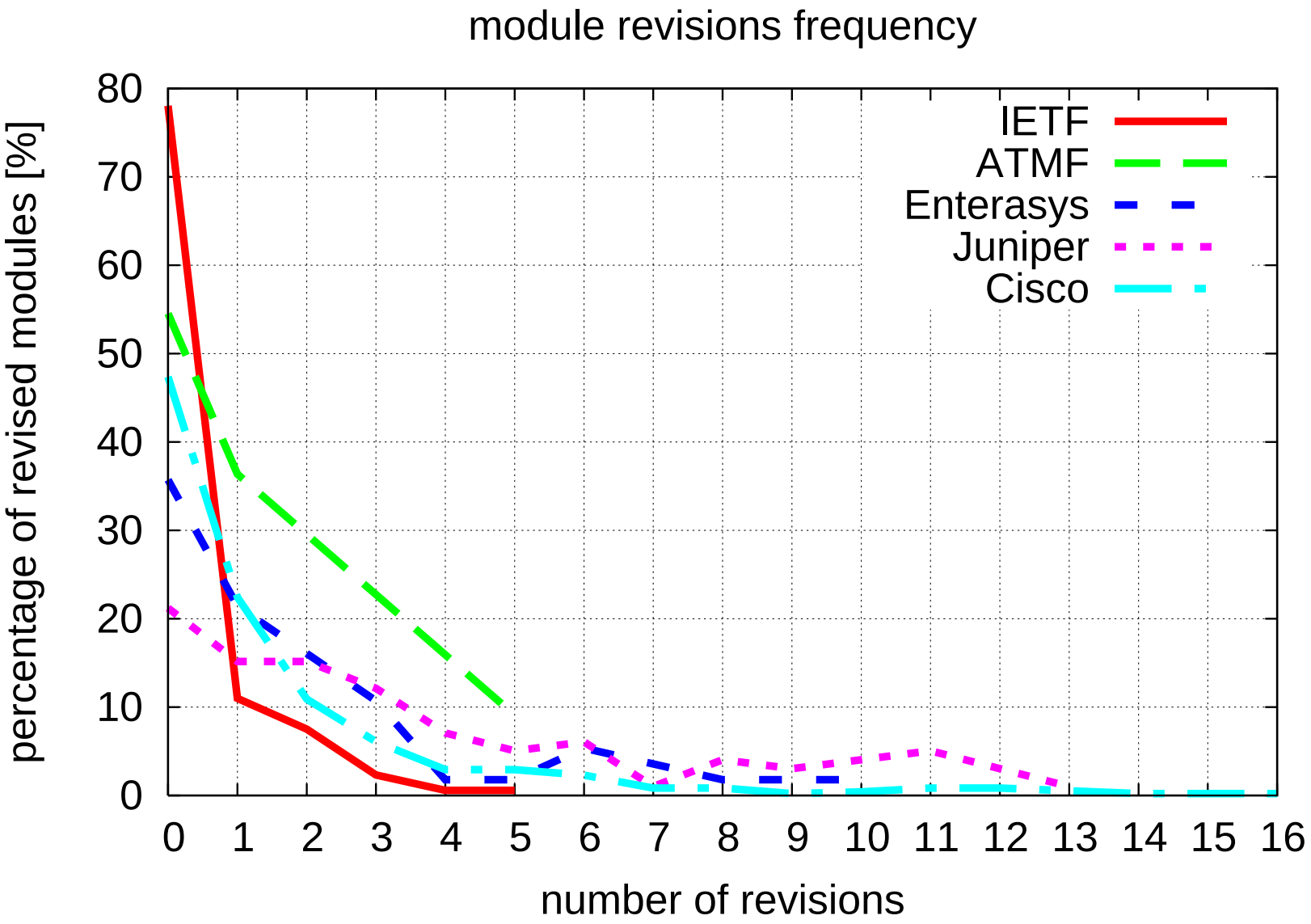
IETF MIB Module Productivity



MIB Module Revision Speed



MIB Module Revision Frequency

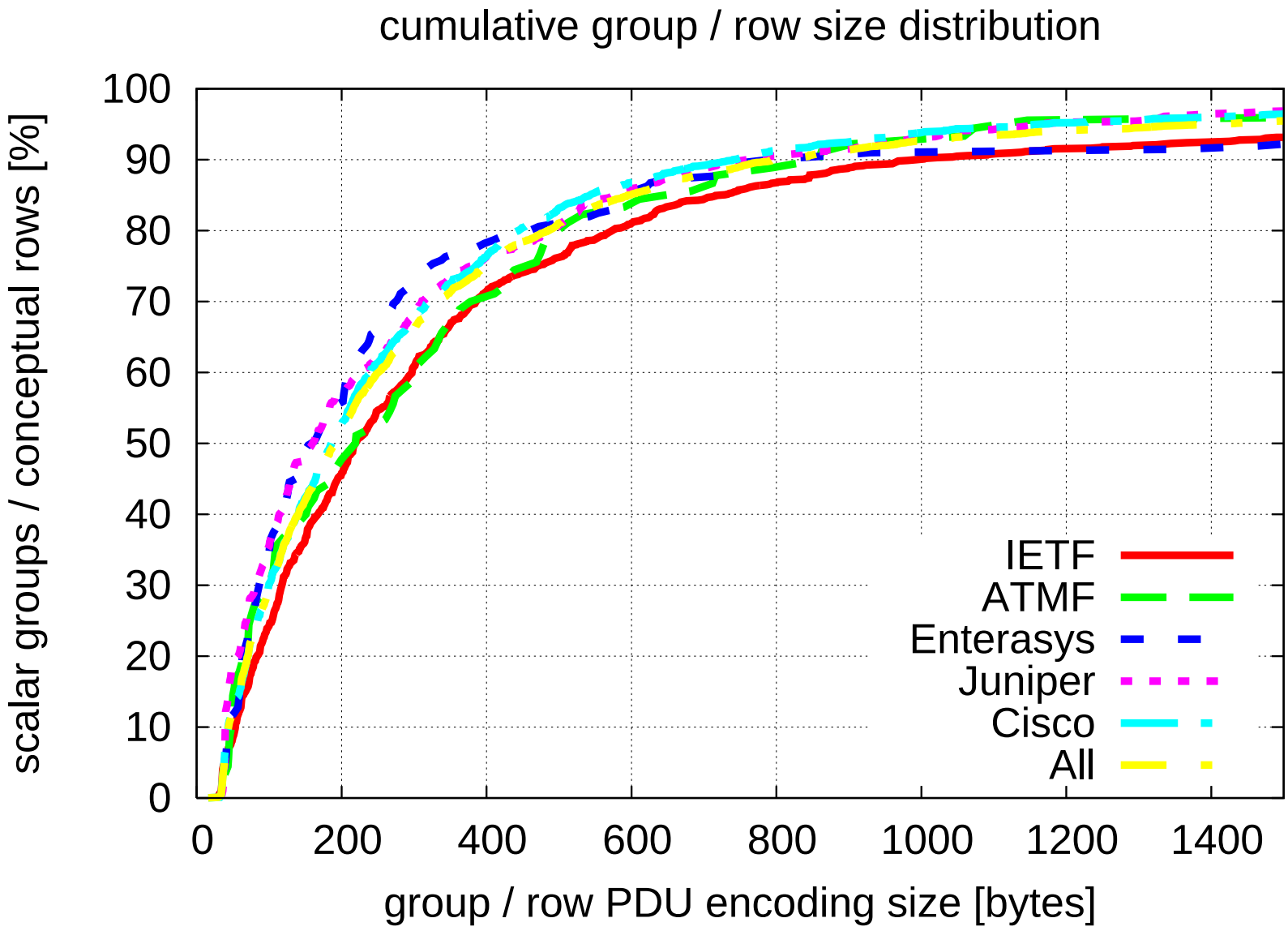


Base Type Usage

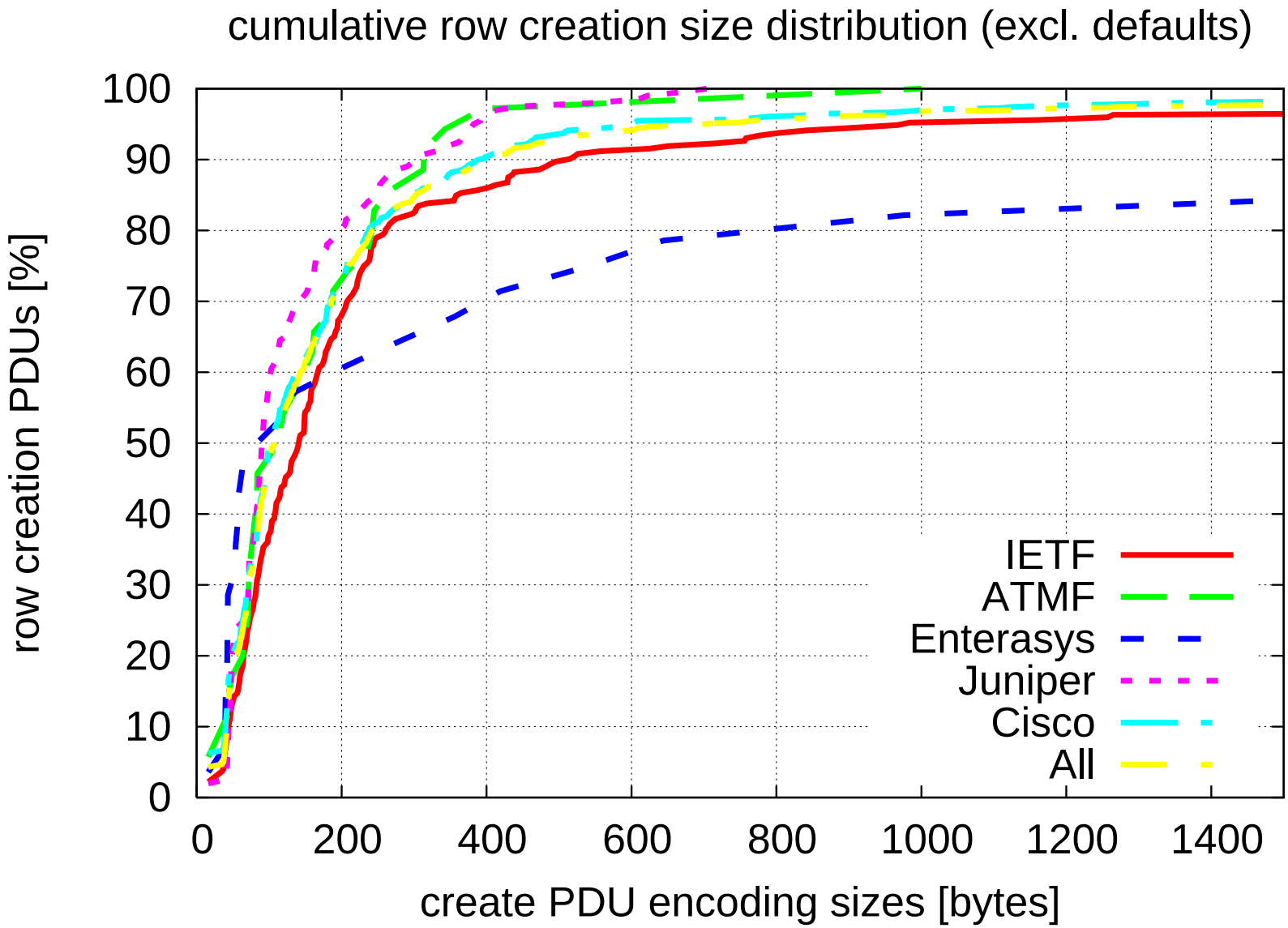
Modules	Int32	Uns32	Uns64	OctetString	ObjectId	Enum	Bits
All	21.5	35.5	3.3	15.0	0.6	23.3	0.9
IETF	22.3	36.5	2.7	16.6	1.9	18.9	1.2
ATM	32.5	27.0	0.0	11.2	0.4	28.1	1.0
Cisco	20.8	38.1	2.8	13.7	0.2	23.6	0.7
Enterasys	18.3	26.7	0.8	22.0	0.2	28.6	3.4
Juniper	21.5	25.6	7.3	17.0	0.2	27.8	0.6

- Looking at all MIB modules, more than 83.6% of all variables are encoded as ASN.1 `INTEGER` values
- Close to 80% are 32-bit integer values that fit into 1-5 bytes
- The actual usage distribution might be different

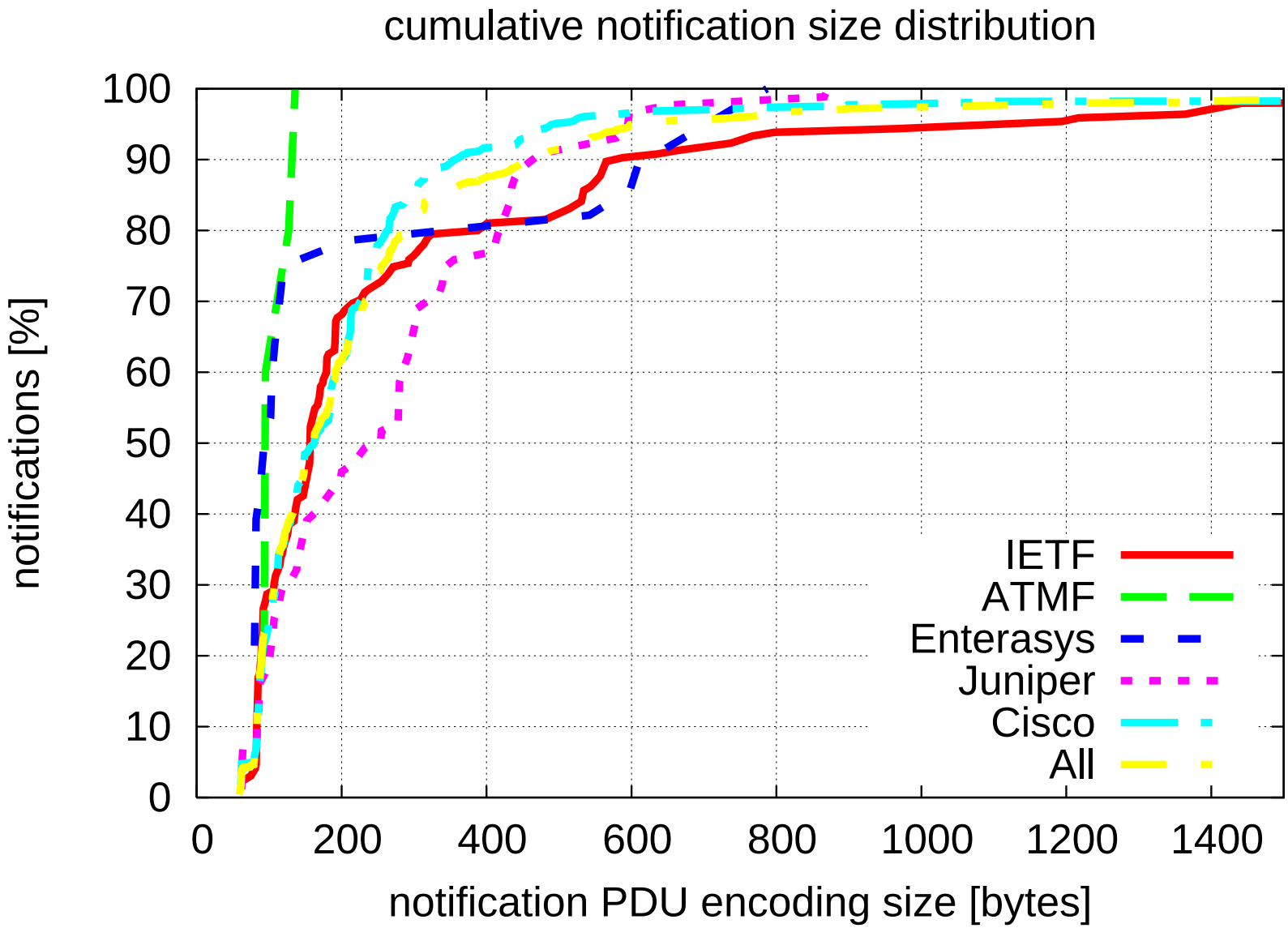
Row Encoding Size Distribution



Create Encoding Size Distribution



Notification Encoding Size Distribution



3. Measurement Approach

Measurement Process

- The measurement process basically consists of five steps:
  1. Capture raw SNMP traces in `pcap` capture files
  2. Convert raw traces into a structured machine and human readable format
  3. Filter converted traffic traces to suppress or anonymize sensitive information
  4. Submit the filtered traces to a repository
  5. Analyze the traces by running analysis scripts
- Note that full packet traces are needed
- Traces may become relatively large files

Division of Work

- There are several approaches to divide the work between network operators and researchers:
  - In some cases, the operator chooses to provide the raw traces to researchers under an NDA
  - In some cases, the operator chooses to provide the filtered / anonymized traces to researchers under an NDA
  - In some cases, the operator chooses to keep the traces under local control and commits to run analysis scripts on them and to provide the results
- To support all approaches, we need to define some common data formats and build tools supporting them

XML Format Example

```
<snmptrace>
  <packet>
    <time-sec>1086871533</time-sec>   <time-usec>344216</time-usec>
    <src-ip>134.169.34.18</src-ip>    <src-port>40776</src-port>
    <dst-ip>134.169.4.162</dst-ip>    <dst-port>161</dst-port>
    <snmp blen="45" vlen="43">
      <version blen="3" vlen="1">0</version>
      <community blen="8" vlen="6">7075626c6963</community>
      <get-next-request blen="32" vlen="30">
        <request-id blen="6" vlen="4">545186112</request-id>
        <error-status blen="3" vlen="1">0</error-status>
        <error-index blen="3" vlen="1">0</error-index>
        <variable-bindings blen="18" vlen="16">
          <varbind blen="16" vlen="14">
            <name blen="12" vlen="10">1.3.6.1.2.1.2.2.1.2.53</name>
            <null blen="2" vlen="0"/>
          </varbind>
        </variable-bindings>
      </get-next-request>
    </snmp>
  </packet>
</snmptrace>
```

XML Format

- Pros:
  - Format captures all SNMP message details
  - Event-driven XML parsers exist for almost all programming languages
  - Human readability allows ad-hoc filtering
  - Formally specified in Relax NG compact notation
- Cons:
  - Trace files can be large, XML files will be even larger
  - Many XML tools/libraries scale badly when XML files are getting large
  - Even efficient XML parsers do add some notable overhead
  - Scripting is not as easy as one might expect since most APIs do not support a notion of a “record”

CSV Format

```
1086871533.344216,134.169.34.18,40776,134.169.4.162,161,45,0,\
get-next-request,545186112,0,0,1,1.3.6.1.2.1.2.2.1.2.53,null,
```

field	description
1	timestamp (in seconds since 1970-01-01)
2-3	source IP address and source port
4-5	destination IP address and port
6	size of the SNMP message
7	protocol version
8	protocol operation
9-11	request-id, error-status, error-index
12	number of varbinds
. . .	for each varbind: a name, a type/exception, a value

CSV Format

- Pros:
  - Implicit filtering of sensitive elements such as community strings
  - Fast to process (a line represents a “record”)
  - Many tools available to manipulate large CSV files
  - Existing scripting APIs are pretty efficient in dealing with CSV files
- Cons:
  - Not all information is captured in the CSV format
  - Format changes cause programs to produce arbitrary garbage

Trace Meta Information

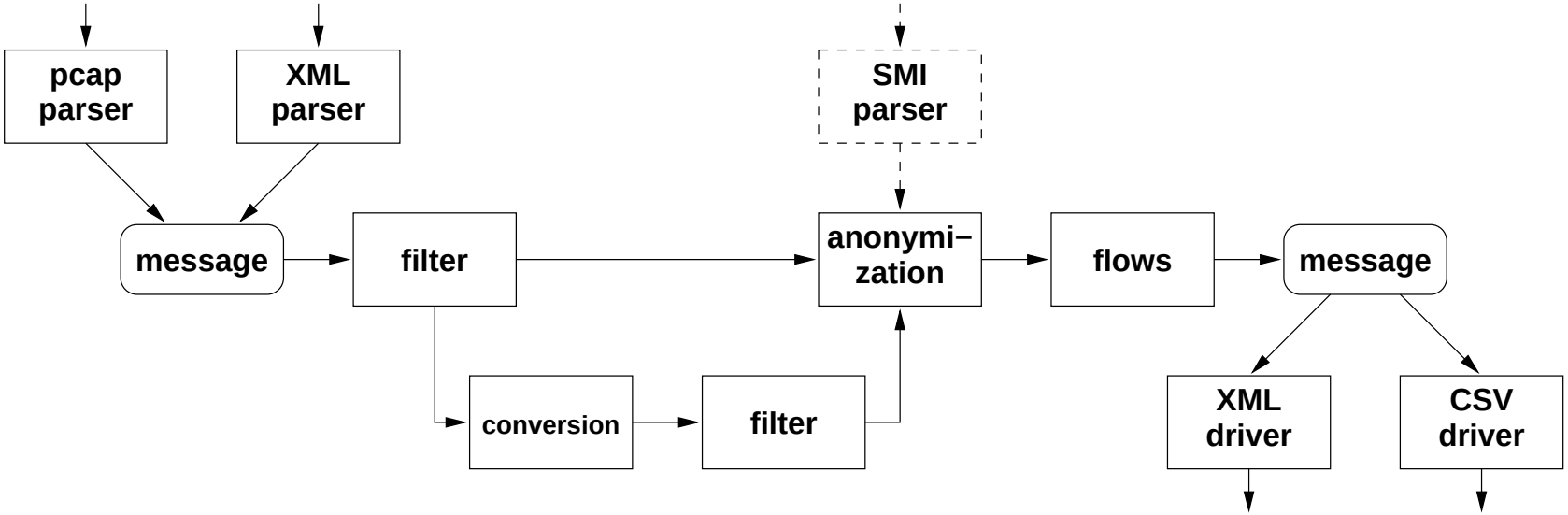
- A simple template has been proposed to record trace meta information:

```
Name:          [name of the trace]
Network:       [name of the network]
Organization:  [name of the organization operating the network]
Contact:       [name and email address of a contact person]
Start-Date:    [date in ISO date format]
End-Date:      [date in ISO date format]
Size:          [size of the pcap trace in bytes]
Description:   [description of the trace]
```

- This might actually be useful to have in XML format...

4. Tool Support

snmpdump and libanon



- The tool `snmpdump` reads as input `pcap` or `xml` files and produces as output `xml` or `csv` files
- The `pcap` parser takes care of reassembly of IP datagrams (`libnids`)
- Internally, SNMP messages are represented in a data structure which associates flags with all data elements

Filtering and Conversion

- The filter module is responsible to remove information that should not be made available (filter-out principle)
  - Filtering must happen as early as possible
  - A filter is a regex matched against field names
  - Typically used to remove community strings
- The conversion module implements trap conversion as specified in RFC 3584 section 3.1
  - Single trap format simplifies scripts
  - Filtering has to be re-applied after conversion
  - Conversion is optional

Anonymization

- The Anonymization module scrambles data in order to raise the effort needed to obtain sensitive information about the internals of an operational network
- Anonymization is rule based:
  - Data matching a given field name or MIB object name or MIB data type is passed to an instance of an anonymization transformation
  - A rule establishes the association between transforms and field names, object names, and data type names
- Appropriate anonymization transforms are implemented as a separate C library called `libanon`

Anonymization (cont.)

- Anonymization transforms typically come in two flavors:
  - Simple randomized substitutions
  - Lexicographic-order preserving substitutions
- For some data types, additional requirements exist:
  - IP address anonymization should preserve the prefix relationships between addresses
  - IEEE 802 broadcast addresses should be preserved and multicast addresses treated separately
  - ...
- Work resulted in a C library of reusable anonymization transforms (`libanon`)
- For details on prefix and lexicographic-order preserving IP address anonymization, see [3]

libanon API

```
anon_ipv4_t* anon_ipv4_new();
void         anon_ipv4_set_key(anon_ipv4_t *a, const anon_key_t *key);
int          anon_ipv4_set_used(anon_ipv4_t *a, const in_addr_t ip,
                                const int prefixlen);
int          anon_ipv4_map_pref(anon_ipv4_t *a, const in_addr_t ip,
                                in_addr_t *aip);
int          anon_ipv4_map_pref_lex(anon_ipv4_t *a, const in_addr_t ip,
                                    in_addr_t *aip);
void         anon_ipv4_delete(anon_ipv4_t *a);

/* key object */

anon_key_t*  anon_key_new();
void         anon_key_set_key(anon_key_t *key, const uint8_t *new_key,
                              const size_t key_len);
void         anon_key_set_random(anon_key_t *key);
void         anon_key_set_passphase(anon_key_t *key,
                                    const char *passphrase);
void         anon_key_delete(anon_key_t *key);
```

Flow Identification

- An SNMP message flow is the collection of SNMP messages between a source and destination address pair which belong to
  - a command generator (CG) / command responder (CR) relationship or
  - a notification originator (NO) / notification receiver (NR) relationship.
- The above definition deliberately does not consider port numbers:
  - Most managed devices include just a single SNMP agent (or they hide multiple agents behind proxies)
  - Management applications often use dynamically allocated port numbers which can change frequently

Analysis Scripts

- Several analysis scripts have been written to
  - extract basic statistics
  - detect walks (sequences of related `getnext`/`getbulk` interactions)
  - analyze the periodic behavior of the traces
  - perform MIB lookups and data aggregations
  - ...
- Some tools are written in Java, others in plain perl
- More work needs to be done...

5. First Results

Initial Set of Traces

loc	short description	period	size
1	national research network	168 hours	5.59 GB
2	university network	246 hours	46.35 GB
3	faculty network	32 hours	1.08 GB
4	server-hosting provider	4 hours	0.006 GB
5	local network provider	21 days	2.2 GB

- Small set of traces used for analysis tool development
- More traces are currently being collected
- Your help is more than welcome: provide data or contacts, especially interesting are special networks types (e.g., DOCSIS networks)

Managers and Agents

loc	packets / min	managers	agents
1	5136	1	178
2	14215	2	224
3	5266	1	27
4	63	3	34
5	711	3	65

- Multi-homed managers are counted multiple times (affects at least location 4)
- Assuming half of the packets contain requests:
  - Loc. 3 agents receive an average of 97 req/m
  - Loc. 4 agents receive no more than 1 req/m
- For location 1 and 3, the number of SYSLOG messages is a fraction of that of SNMP

SNMP Versions

loc	SNMPv1	SNMPv2c	SNMPv3
1	100%	0%	0%
2	5.5%	94.5%	0%
3	11.4%	88.6%	0%
4	35.7%	64.3%	0%
5	100%	0%	0%

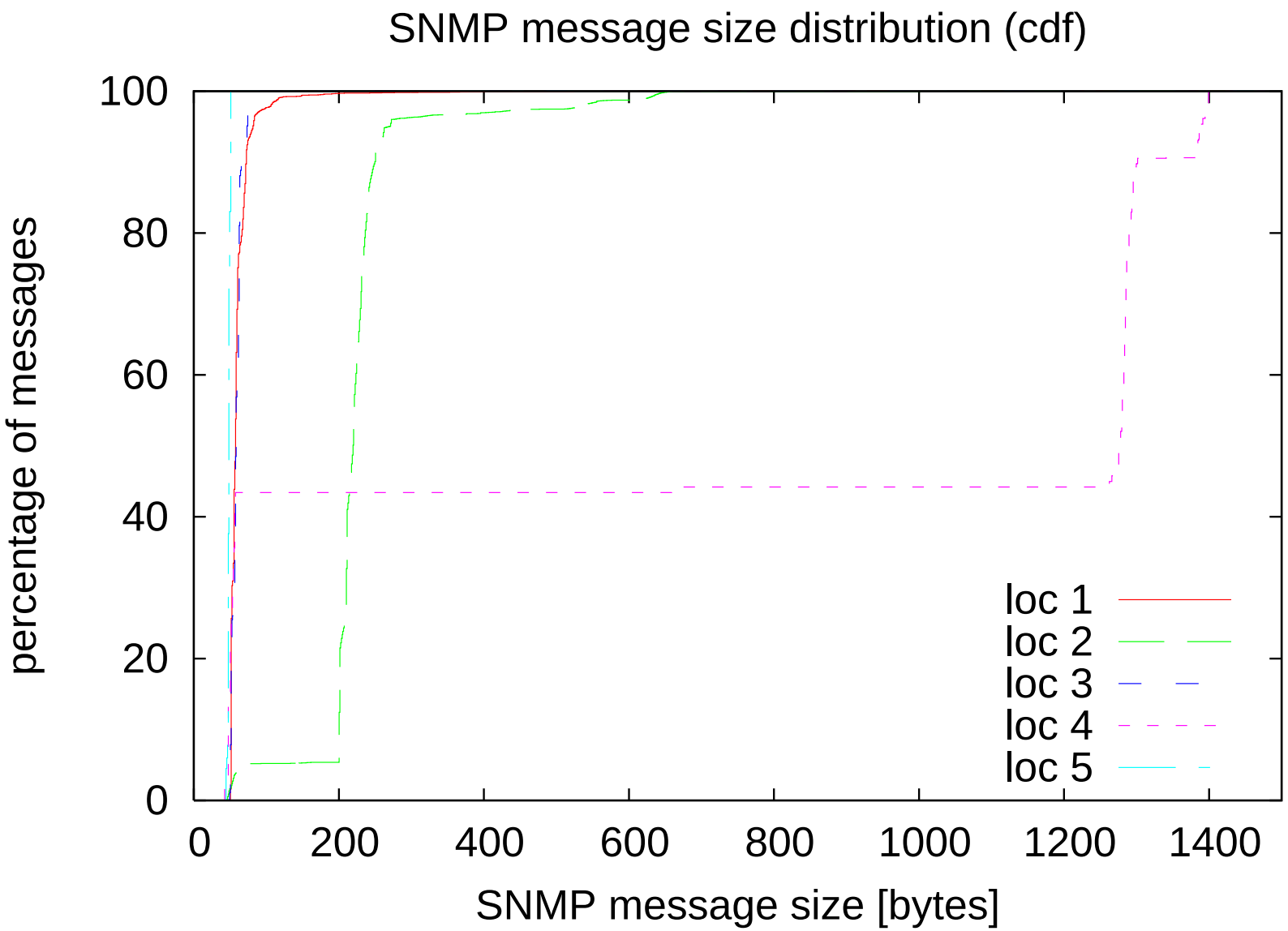
- SNMPv1 and SNMPv2c are popular in these traces

Protocol Operations

loc	Get	GetNext	GetBulk	Trap	Response
1	0.04%	50.0%	0.0%	0.0%	50.0%
2	0.1%	2.4%	47.1%	0.7%	49.6%
3	5.7%	44.3%	0.0%	0.01%	50.0%
4	32.8%	3.8%	22.9%	0.0%	40.5%
5	50.0%	0.0%	0.0%	0.0%	50.0%

- The relative low number of responses in location 4 were due to some systems switched off for maintenance
- Some sites use SNMPv2c without using SNMPv2c protocol operations (see next slide)

Message Size Distribution



Popular MIB Modules

MIB module	loc 1	loc 2	loc 3	loc 4	loc 5
RFC1213-MIB	10.32	0.01	0.02	0	6.64
IF-MIB	40.31	97.23	31.48	61.52	26.70
IP-MIB	0.14	0.06	0.00	0	0
BGP4-MIB	17.55	0	0.00	0	0
SNMPv2-MIB	0.04	0.79	0.00	6.56	0.00
BRIDGE-MIB	0.00	1.60	66.65	0	0
Unknown	31.63	0.32	1.85	31.91	66.66

- Popular enterprise specific MIB modules in these traces are from Cisco Systems, Hewlett-Packard, and APC (a UPS manufacturer)

6. Conclusions

Conclusions

- Work to develop models of SNMP traffic has started
- Core tools and analysis scripts have been developed
- More traces are needed (perhaps you can help?)
- More analysis scripts are written (you can help again)
- Work may evolve to look beyond SNMP and to cover also other management protocols (SYSLOG, CLIs, . . .)

Thanks!

References

[1] J. Schönwälder. Characterization of SNMP MIB Modules. In *Proc. 9th IFIP/IEEE International Symposium on Integrated Network Management*, pages 615–628. IEEE, May 2005.

[2] J. Schönwälder. SNMP Traffic Measurements. Internet Draft <draft-irtf-nmrg-snmp-measure-00.txt>, International University Bremen, May 2006.

[3] M. Harvan and J. Schönwälder. Prefix- and Lexicographical-order-preserving IP Address Anonymization. In *10th IEEE/IFIP Network Operations and Management Symposium*, April 2006.

[4] A. Pras, J. Schönwälder, M. Harvan, J. Schippers, and R. van de Meent. SNMP Traffic Analysis: Approaches, Tools, and First Results. In *under preparation*.