Capturing and Processing One Million Network Flows Per Second with SiLK: Challenges and Strategies

Robert W. Techentin

David R. Holmes, III

James C. Nelms

Barry K. Gilbert

Presented to FloCon 2016, Daytona Beach, FL

January 12, 2016

Outline

- Description of the Environment
- Extract / Transform / Load Process Pipeline
- Performance Challenges for SiLK flowcap
- Parallel flowcap processing
- De-Duplication
- Summarization and Translation
- Implementation Details

Teamwork

- Special Purpose Processor Development Group
  - Barry Gilbert, Ph.D.
- Biomedical Analytics and Computational Engineering
  - David R. Holmes, III, Ph.D.
- Office of Information Security
  - James C. Nelms

Will and Charlie Mayo, The Mayo Brothers

MAYO CLINIC

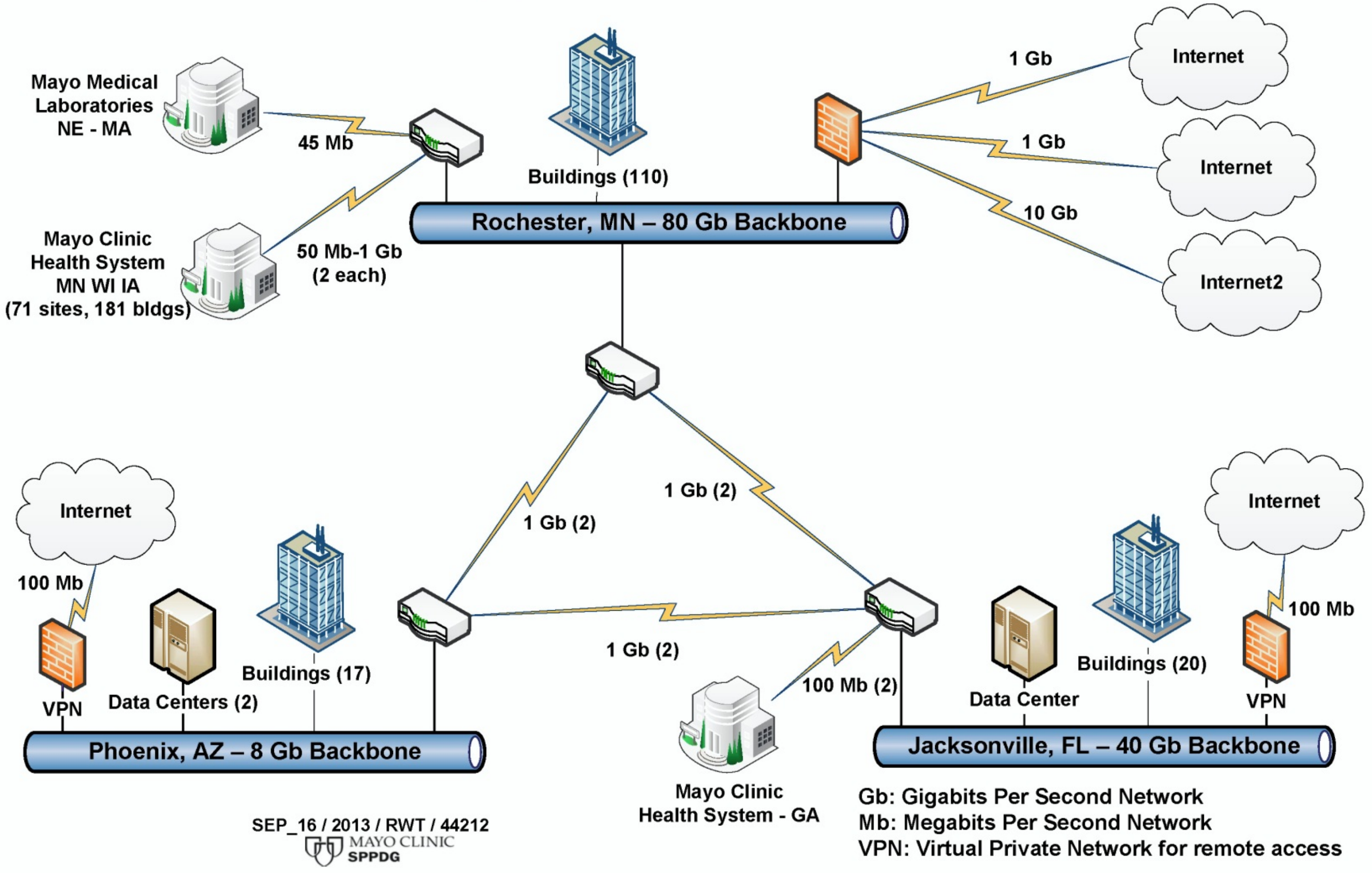
Mayo Clinic Networking

- Mayo Clinic is a substantial enterprise
  - More than 60,000 employees; 1.1 million patients per year
  - Clinical practice, research and education
  - Spans seven states, hundreds of buildings
  - And on top of the “usual” business issues (e.g., intellectual property), medical centers must comply with HIPAA regulations
- Mayo’s computer network and applications are large and complex
  - Commercial, clinical, and business IT applications
  - Medical equipment, custom systems, and applications
  - Thousands of routers and network devices
  - Hundreds of thousands of IP addresses

MAYO CLINIC

MAYO CLINIC NETWORK OVERVIEW

(117,000 Networked Devices; 370 Routers; 4,300 Switches; 5,600 Wireless Access Points; Spanning 330 Buildings in 7 States; Over 55,000 Employees; 1.1 Million Patient Visitors Per Year)



Mayo Medical Laboratories NE - MA
45 Mb
Mayo Clinic Health System MN WI IA (71 sites, 181 bldgs)
50 Mb-1 Gb (2 each)
Buildings (110)
Rochester, MN – 80 Gb Backbone
1 Gb
Internet
1 Gb
Internet
10 Gb
Internet2
1 Gb (2)
1 Gb (2)
1 Gb (2)
Internet
100 Mb
VPN
Data Centers (2)
Buildings (17)
Phoenix, AZ – 8 Gb Backbone
100 Mb (2)
Mayo Clinic Health System - GA
Data Center
Buildings (20)
Internet
100 Mb
VPN
Jacksonville, FL – 40 Gb Backbone

Gb: Gigabits Per Second Network
Mb: Megabits Per Second Network
VPN: Virtual Private Network for remote access

SEP_16 / 2013 / RWT / 44212
MAYO CLINIC
SPPDG

Defending the Mayo Network

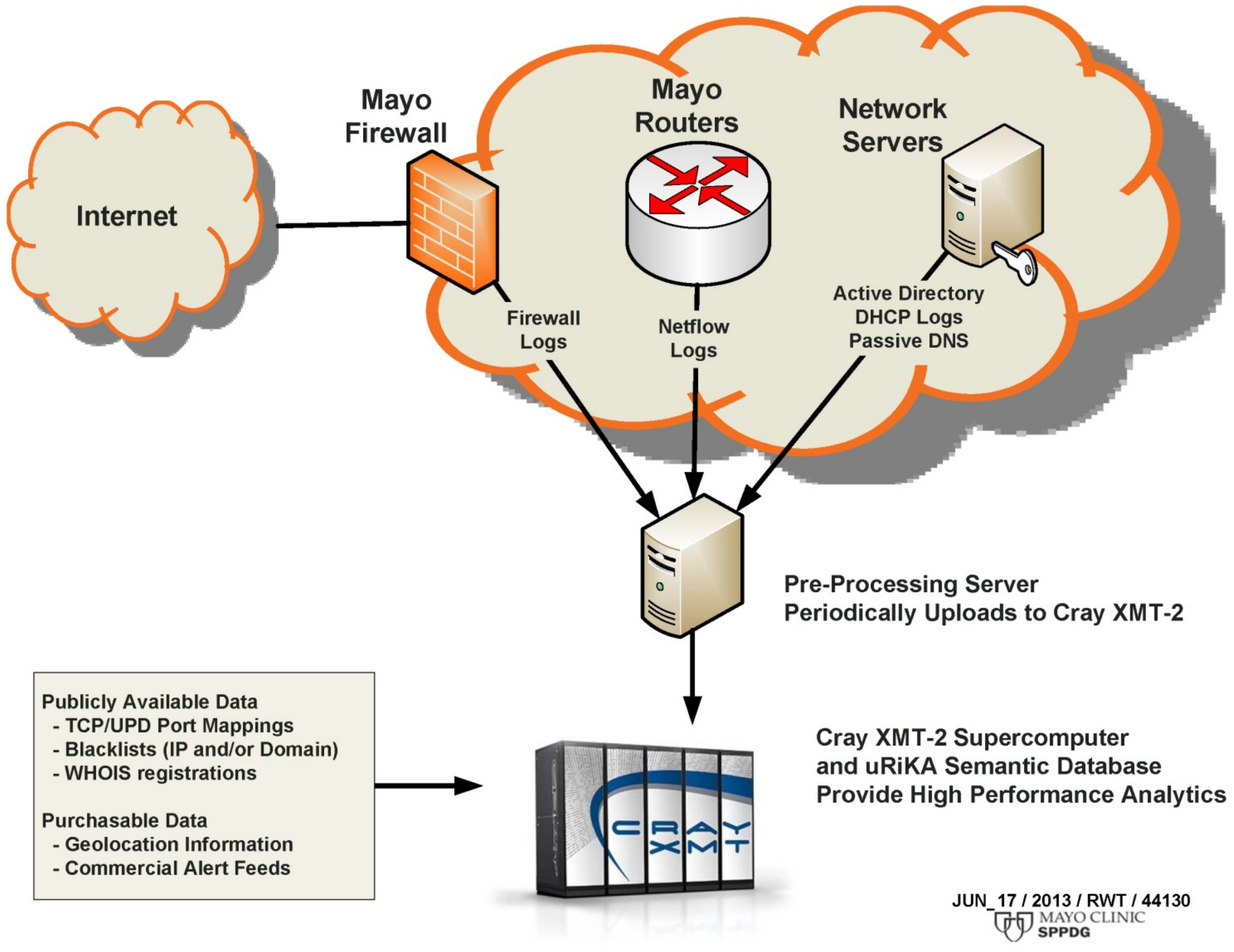
- Network defense is only one of the missions of the Office of Information Security (OIS)
- Traditional network defense technologies
  - Firewall; White/Black Listing; Deep Packet Inspection
- Threat Response Center
- Threat Intelligence Team
- Training and involvement for all employees
- Development of advanced capabilities to gain an edge on attackers

MAYO CLINIC

Mayo Office of Information Technology Vision for Advanced Network Analytics

- Develop advanced analytic tools to support the current and future OIS mission
- Focus on identity resolution, behavior classification, and anomalous events
- Exploit emerging algorithms and capabilities of graph analytics (not commonly employed in commercial solutions)
- Scale to entire Mayo wide area network and all business activities
    - Exploit graph supercomputer as a target of opportunity

PROPOSED CYBER SECURITY ANALYSIS USING CRAY XMT-2 SUPERCOMPUTER (STAGE 1)



Publicly Available Data
- TCP/UPD Port Mappings
- Blacklists (IP and/or Domain)
- WHOIS registrations

Purchasable Data
- Geolocation Information
- Commercial Alert Feeds

JUN_17 / 2013 / RWT / 44130

MAYO CLINIC
SPPDG

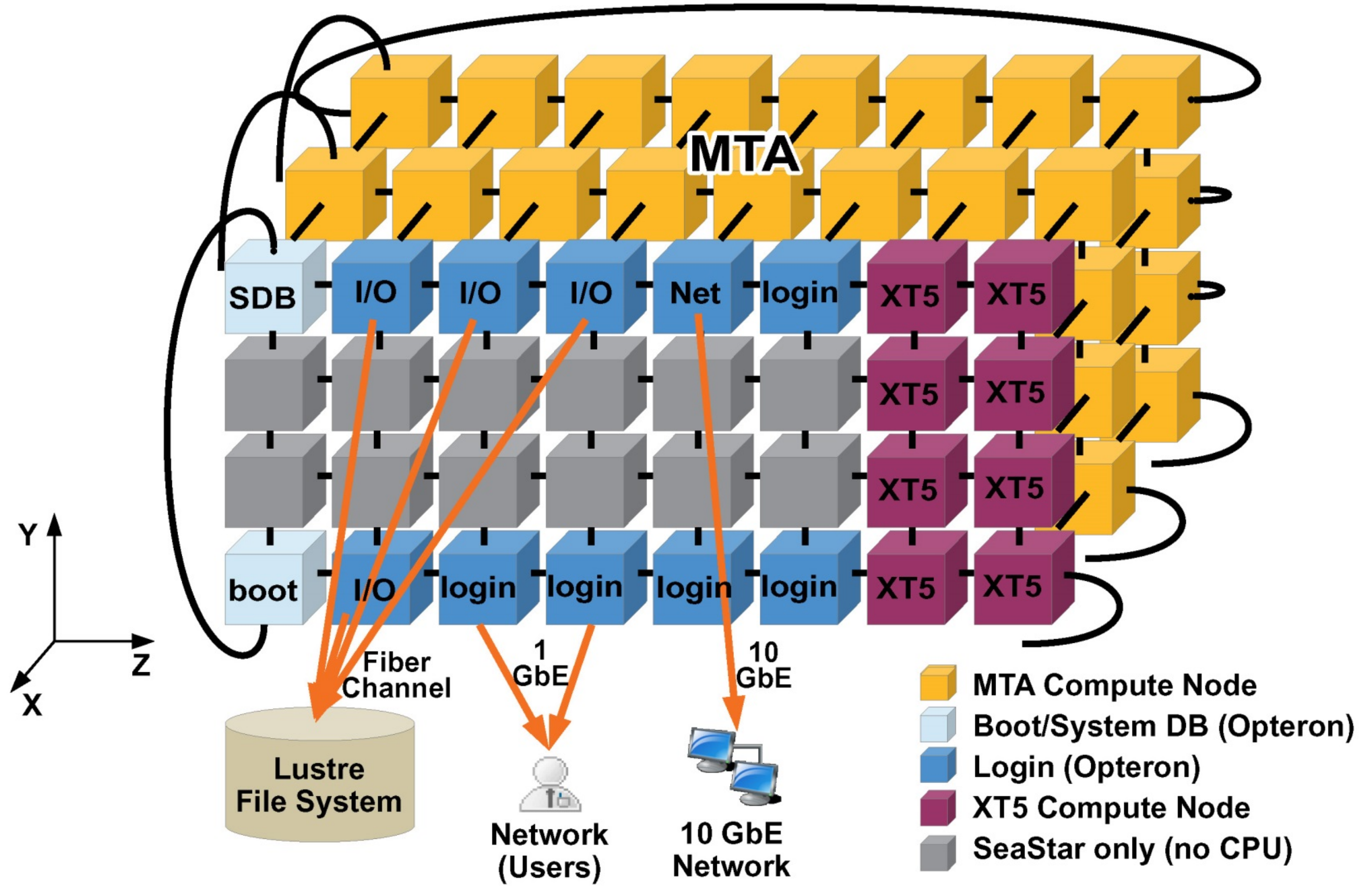
CRAY XMT-2 SYSTEM "GRACE" INSTALLED AT MAYO SUPPORT CENTER IN OCTOBER 2011 (Single Compute Cabinet; 64 Threadstorm Processors Run Up to 8192 Threads; 2 TB Dynamic RAM; 27 TB Fast RAID Storage)

OCT_19 / 2011 / RWT / 43000

MAYO CLINIC
SPPDG

HYBRID XMT-2 SYSTEM PROCESSOR AND INTERNAL NETWORK CONFIGURATION

(Hybrid XMT-2 System Named "Grace" at Mayo Clinic; MTA Multithreaded Architecture with 64 Processors, 2 TB Shared DRAM, and 8192 Threads; XT5 Compute Cluster of 8 Nodes, Each with 32 GB DRAM and 12 Cores; Twelve Service Nodes Provide Login, I/O, and Network Support)



🟧	MTA Compute Node
⬜	Boot/System DB (Opteron)
🟦	Login (Opteron)
🟪	XT5 Compute Node
⬛	SeaStar only (no CPU)

Routing is deterministic, static, and determined at boot time, and by default follows first the X dimension, then Y, then Z. (Note: Not all torus links shown)

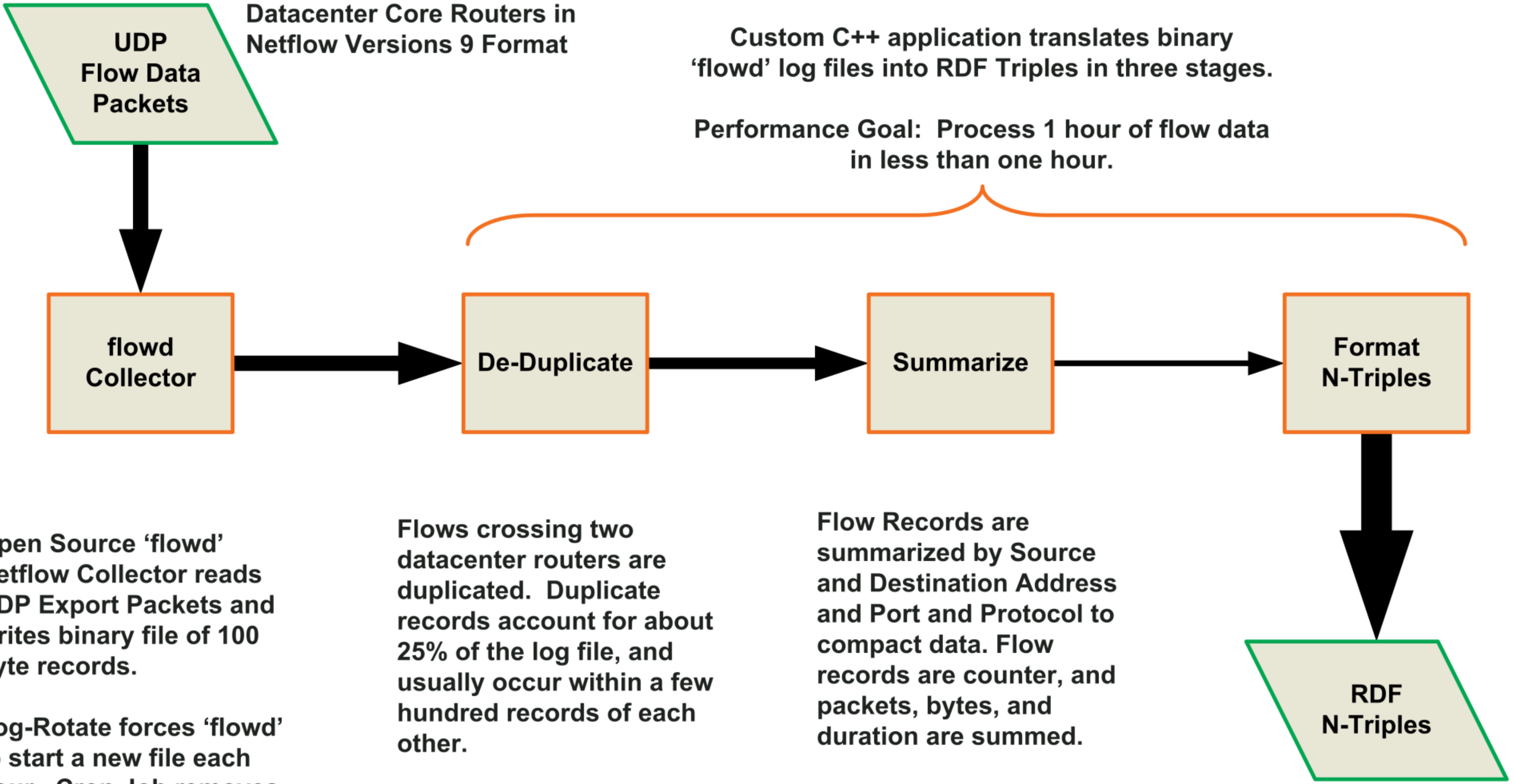
I/O = Lustre I/O Node
Net = 10 GbE Network Node

JUL_03 / 2013 / RWT / 44157
MAYO CLINIC
SPPDG

Extract / Transform / Load Network Data

- The first step in analyzing the network is capturing data
- Many different forms of data are available, including Netflow, DNS requests, syslog events, network topology, asset and user databases
- However, the largest and most intractable data source is Netflow
  - Mayo Clinic network reports "full take" of all Netflow records through a hierarchy of concentrators
  - Capturing, formatting, and loading data is challenging, even for "near real time" performance

FIRST GENERATION NETFLOW EXTRACT-TRANSFORM-LOAD SOFTWARE ARCHITECTURE FOR CONVERTING FLOW RECORDS INTO RESOURCE DESCRIPTION FRAMEWORK (RDF) N-TRIPLE FORMAT SUMMARY DATA



Incoming Flow Data from Nine Datacenter Core Routers in Netflow Versions 9 Format

Custom C++ application translates binary 'flowd' log files into RDF Triples in three stages.

Performance Goal: Process 1 hour of flow data in less than one hour.

Open Source 'flowd' Netflow Collector reads UDP Export Packets and writes binary file of 100 byte records.

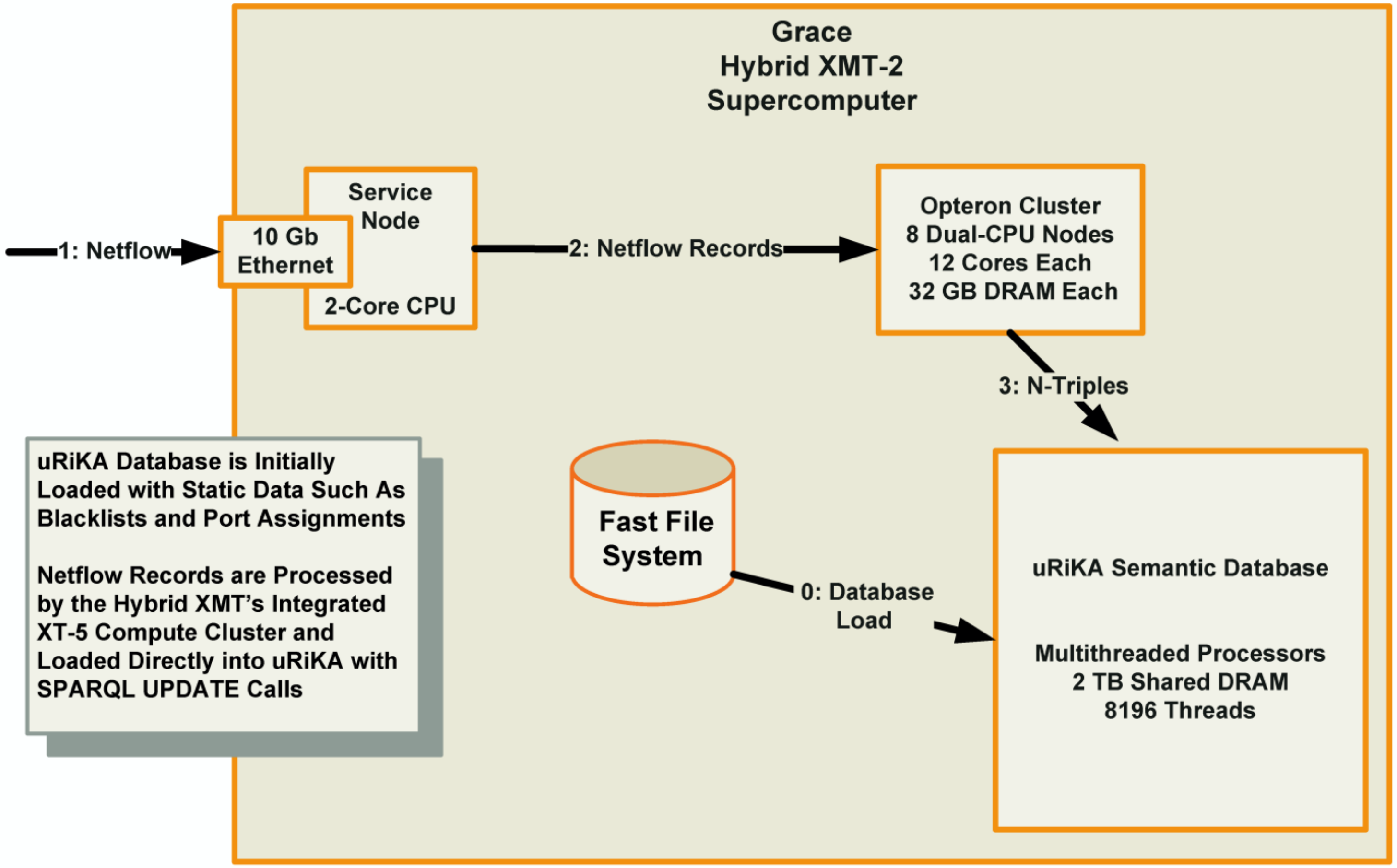
Log-Rotate forces 'flowd' to start a new file each hour. Cron Job removes old files after one day. Files are typically 30 GB.

Flows crossing two datacenter routers are duplicated. Duplicate records account for about 25% of the log file, and usually occur within a few hundred records of each other.

Flow Records are summarized by Source and Destination Address and Port and Protocol to compact data. Flow records are counter, and packets, bytes, and duration are summed.

Conversion to the extremely verbose N-Triple format expands the hourly data files to 85 GB.

PROPOSED DATA FLOW FOR SECOND PHASE OF IMPROVED NETWORK TRAFFIC DATA CAPTURE INTO XMT-2 uRiKA SYSTEM FOR CYBER ANALYSIS
(Netflow Records are Forwarded to Integrated XT-5 Compute Cluster for Parallel Conversion to RDF Triples; Triples are Loaded Directly into uRiKA)



uRiKA Database is Initially Loaded with Static Data Such As Blacklists and Port Assignments

Netflow Records are Processed by the Hybrid XMT's Integrated XT-5 Compute Cluster and Loaded Directly into uRiKA with SPARQL UPDATE Calls

DEC_12 / 2013 / RWT / 44331
MAYO CLINIC
SPPDG

Netflow Extract / Transform / Load Process: First and Second Generations

- First generation ETL
  - Capture Netflow v9 from datacenter core routers
  - "Full take" up to 200K records per second
    - All flows entering or crossing the datacenters
  - No real time requirements
- Second generation ETL – bigger and more complex
  - Netflow versions 5, 9, and IPFIX (v10)
  - "Full take" from thousands of network devices
  - Anticipate up to 1 million flows per second, peak
  - Near-real-time desirable

MAYO CLINIC

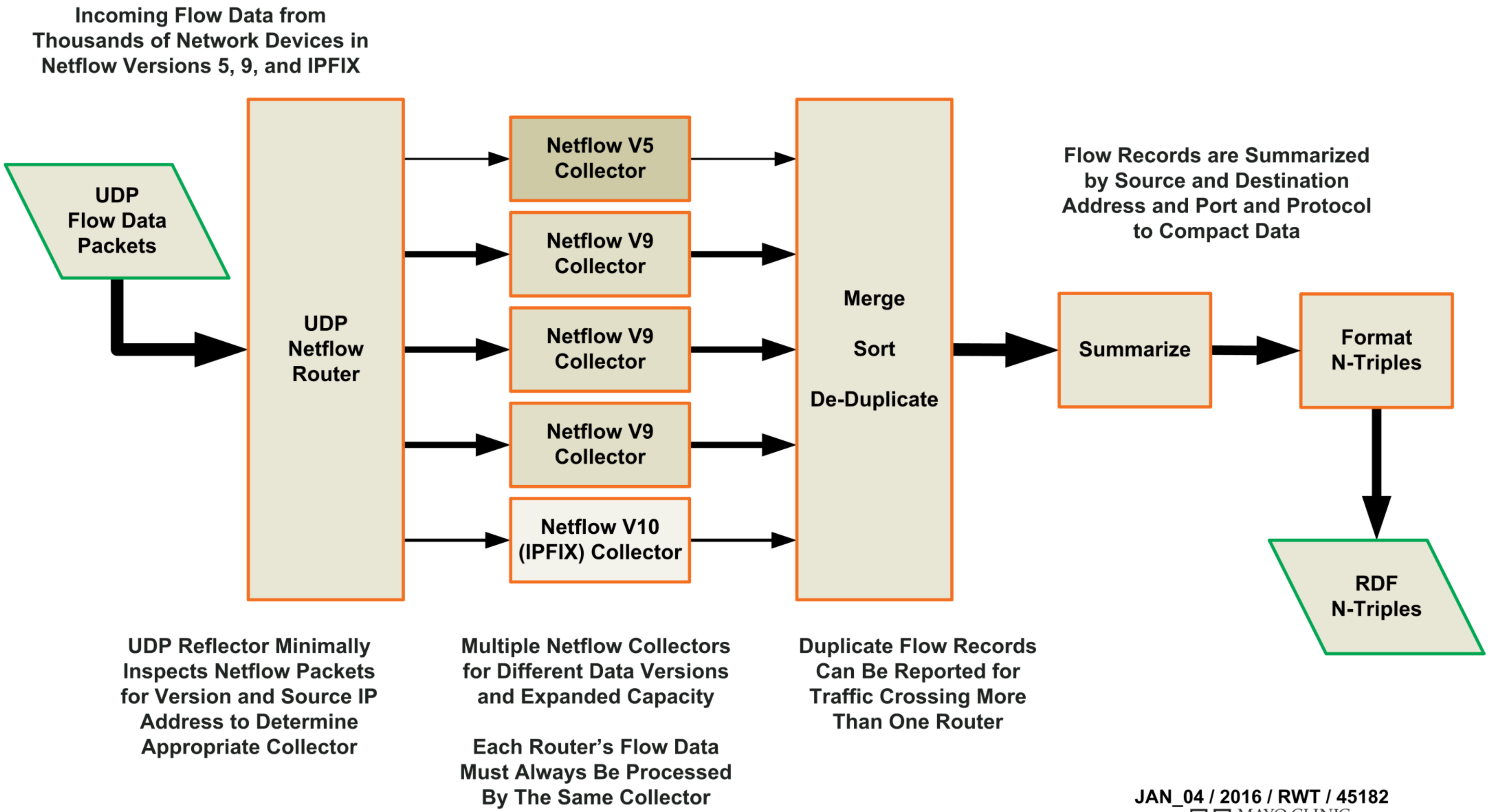
Evaluation of Open Source Netflow Collectors

- First generation implemented with ‘flowd’
  - Fast enough for 300K records per second
  - Limited to Netflow v5 and V9
- SiLK flowcap
  - Captures Netflow v5, v9, and IPFIX (v10)
  - Many associated tools for pipeline processing
- nfdump / nfcapd
  - Includes filtering, aggregation and printf() formatting
  - Experimental IPFIX support
  - Recommends one process per netflow source

SiLK Flowcap Limitations

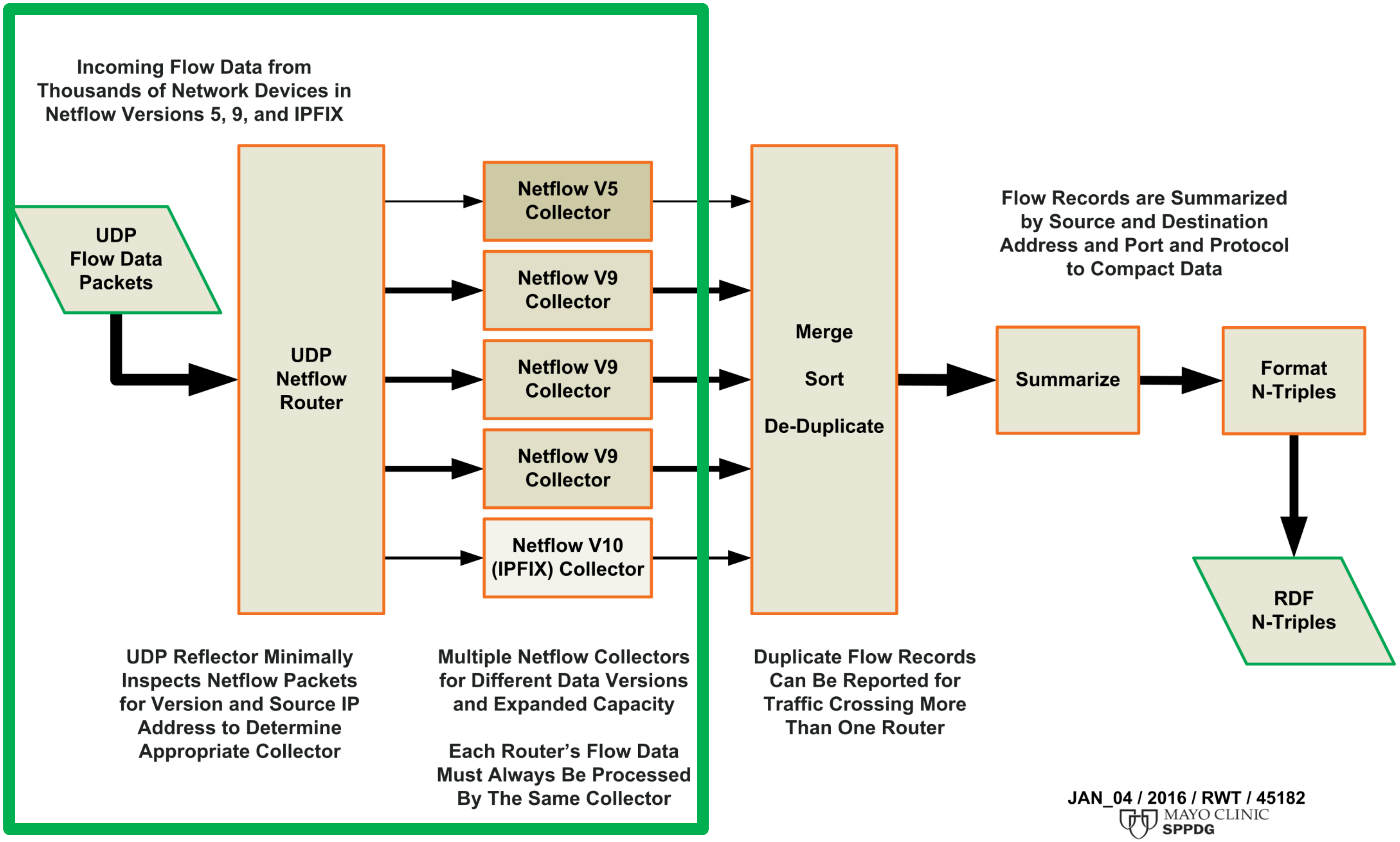
- One instance of flowcap can comprehend only one version of Netflow
  - Requires multiple instances of flowcap
  - Each instance ignores packets that it cannot interpret
    - However, performance impact is unknown
- Flowcap performance likely cannot support one million flows per second
  - Requires parallel processing
  - Which requires intelligent splitting of the Netflow stream
    - Each v9 and IPFIX router sends templates for its flow data
    - A flowcap instance must receive both templates and flow records

SECOND GENERATION NETFLOW EXTRACT-TRANSFORM-LOAD SOFTWARE ARCHITECTURE FOR PARALLEL PROCESSING FLOW RECORDS INTO RESOURCE DESCRIPTION FRAMEWORK (RDF) N-TRIPLE FORMAT SUMMARY DATA


Incoming Flow Data from Thousands of Network Devices in Netflow Versions 5, 9, and IPFIX
UDP Flow Data Packets
UDP Netflow Router
Netflow V5 Collector
Netflow V9 Collector
Netflow V9 Collector
Netflow V9 Collector
Netflow V10 (IPFIX) Collector
Merge
Sort
De-Duplicate
Flow Records are Summarized by Source and Destination Address and Port and Protocol to Compact Data
Summarize
Format N-Triples
RDF N-Triples
UDP Reflector Minimally Inspects Netflow Packets for Version and Source IP Address to Determine Appropriate Collector
Multiple Netflow Collectors for Different Data Versions and Expanded Capacity
Each Router's Flow Data Must Always Be Processed By The Same Collector
Duplicate Flow Records Can Be Reported for Traffic Crossing More Than One Router

JAN_04 / 2016 / RWT / 45182
MAYO CLINIC
SPPDG

SECOND GENERATION NETFLOW EXTRACT-TRANSFORM-LOAD SOFTWARE ARCHITECTURE FOR PARALLEL PROCESSING FLOW RECORDS INTO RESOURCE DESCRIPTION FRAMEWORK (RDF) N-TRIPLE FORMAT SUMMARY DATA


Incoming Flow Data from Thousands of Network Devices in Netflow Versions 5, 9, and IPFIX
UDP Flow Data Packets
UDP Netflow Router
Netflow V5 Collector
Netflow V9 Collector
Netflow V9 Collector
Netflow V9 Collector
Netflow V10 (IPFIX) Collector
Merge
Sort
De-Duplicate
Flow Records are Summarized by Source and Destination Address and Port and Protocol to Compact Data
Summarize
Format N-Triples
RDF N-Triples
UDP Reflector Minimally Inspects Netflow Packets for Version and Source IP Address to Determine Appropriate Collector
Multiple Netflow Collectors for Different Data Versions and Expanded Capacity
Each Router's Flow Data Must Always Be Processed By The Same Collector
Duplicate Flow Records Can Be Reported for Traffic Crossing More Than One Router

JAN_04 / 2016 / RWT / 45182
MAYO CLINIC
SPPDG

Separating Flow Versions with UDP Reflector

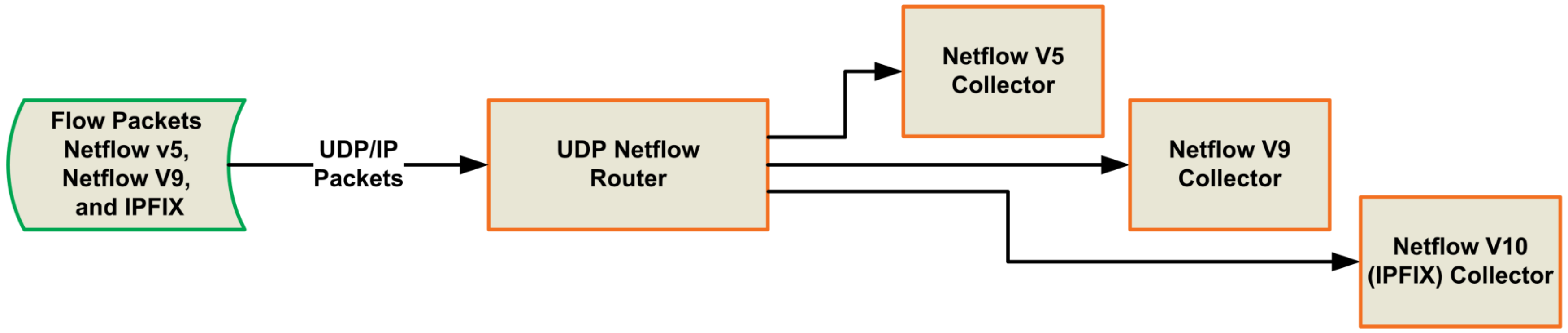
- We considered netcat and iptables to replicate flow packets to multiple destinations
  - However, duplicated flow data consumes network resources and may impact collector performance
  - And the available version of iptables did not support TEE
- UDP Reflector (https://code.google.com/p/udp-reflector/) provided framework for intelligent routing of flow packets
  - Supports multiple packet destinations and filtering
  - Uses libpcap – very fast and below the IP stack
  - Source code available for modification

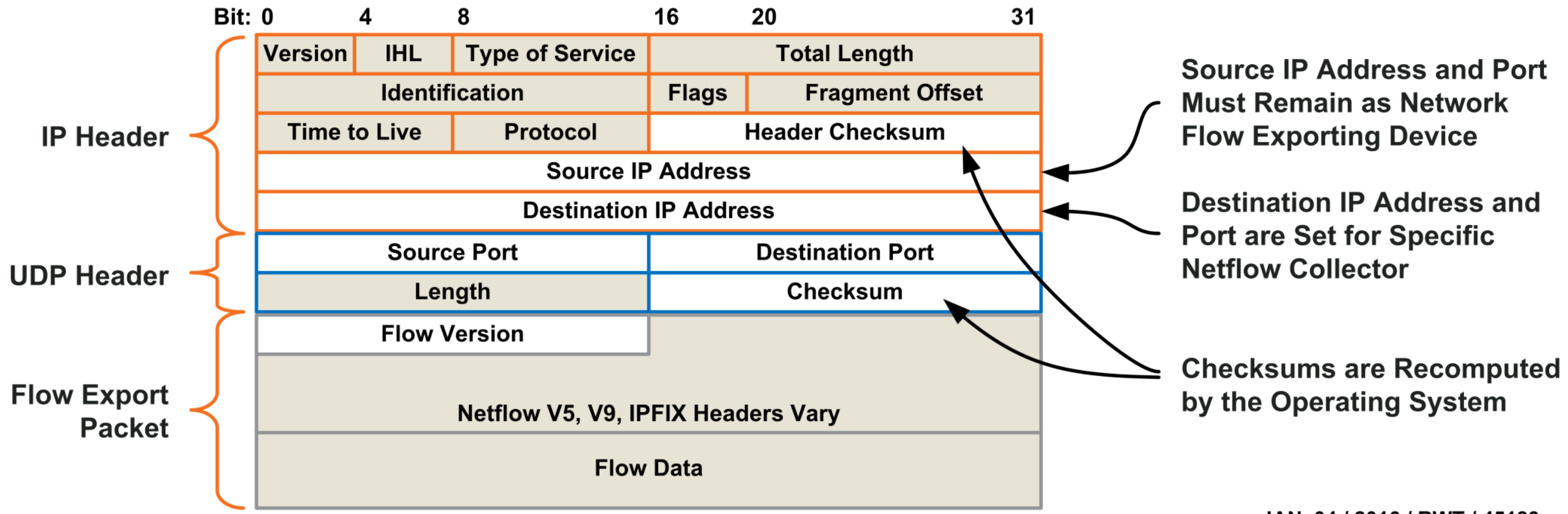
Custom UDP Netflow Router

- Customized UDP Reflector code base
  - Listens on specific UDP port
  - Captures packets with libpcap
- Inspects packets for Netflow version field
  - Chooses specific Netflow collector
- Re-writes destination address and port for specific collector
- Ensures that source address matches originating exporting device
- Recomputes checksums
- Forwards packets to collector

ROUTING NETFLOW EXPORT PACKETS REQUIRES MANIPULATION OF UDP AND IP HEADERS AS WELL AS PARSING NETFLOW VERSION NUMBER FROM EXPORT PACKET HEADER

(Custom Netflow Router Application Based on Open Source "UDP Reflector")


Flow Packets Netflow v5, Netflow V9, and IPFIX
UDP/IP Packets
UDP Netflow Router
Netflow V5 Collector
Netflow V9 Collector
Netflow V10 (IPFIX) Collector


Netflow Export Packet Encapsulated in UDP and IP Protocols
Bit: 0 4 8 16 20 31
IP Header
Version
IHL
Type of Service
Total Length
Identification
Flags
Fragment Offset
Time to Live
Protocol
Header Checksum
Source IP Address
Destination IP Address
UDP Header
Source Port
Destination Port
Length
Checksum
Flow Export Packet
Flow Version
Netflow V5, V9, IPFIX Headers Vary
Flow Data
Source IP Address and Port Must Remain as Network Flow Exporting Device
Destination IP Address and Port are Set for Specific Netflow Collector
Checksums are Recomputed by the Operating System

JAN_04 / 2016 / RWT / 45183

MAYO CLINIC SPPDG

Flowcap Performance Measurement

- Flowcap capacity was estimated* to be 100-300K flows/sec
  - Each Netflow version supported by different code base
  - Hardware and OS and network stack add variability
- Needed capture/replay capability for Netflow export packets
  - Different from YAF, which converts pcap data to IPFIX
  - “tcpreplay” did not work correctly on network service nodes
- Constructed custom record / replay application
  - Based on UDP Reflector
  - Replay speed variable by simple inter-packet delay

* netsa-discuss mailing list and private emails

Results of Flowcap Performance Tests

- Recorded live Netflow packet streams (up to 20 minutes)
- Replayed packet stream to flow collector
    - Starting with “long” inter-packet delay, and recorded collector results
    - Slowly decreased delay, checking for dropped flows
- Computed collector “flows per second” (fps, or kfps) as number of flow records divided by minimum playback time

- Flowcap for v9 reliably achieved 100 kfps (for this system)

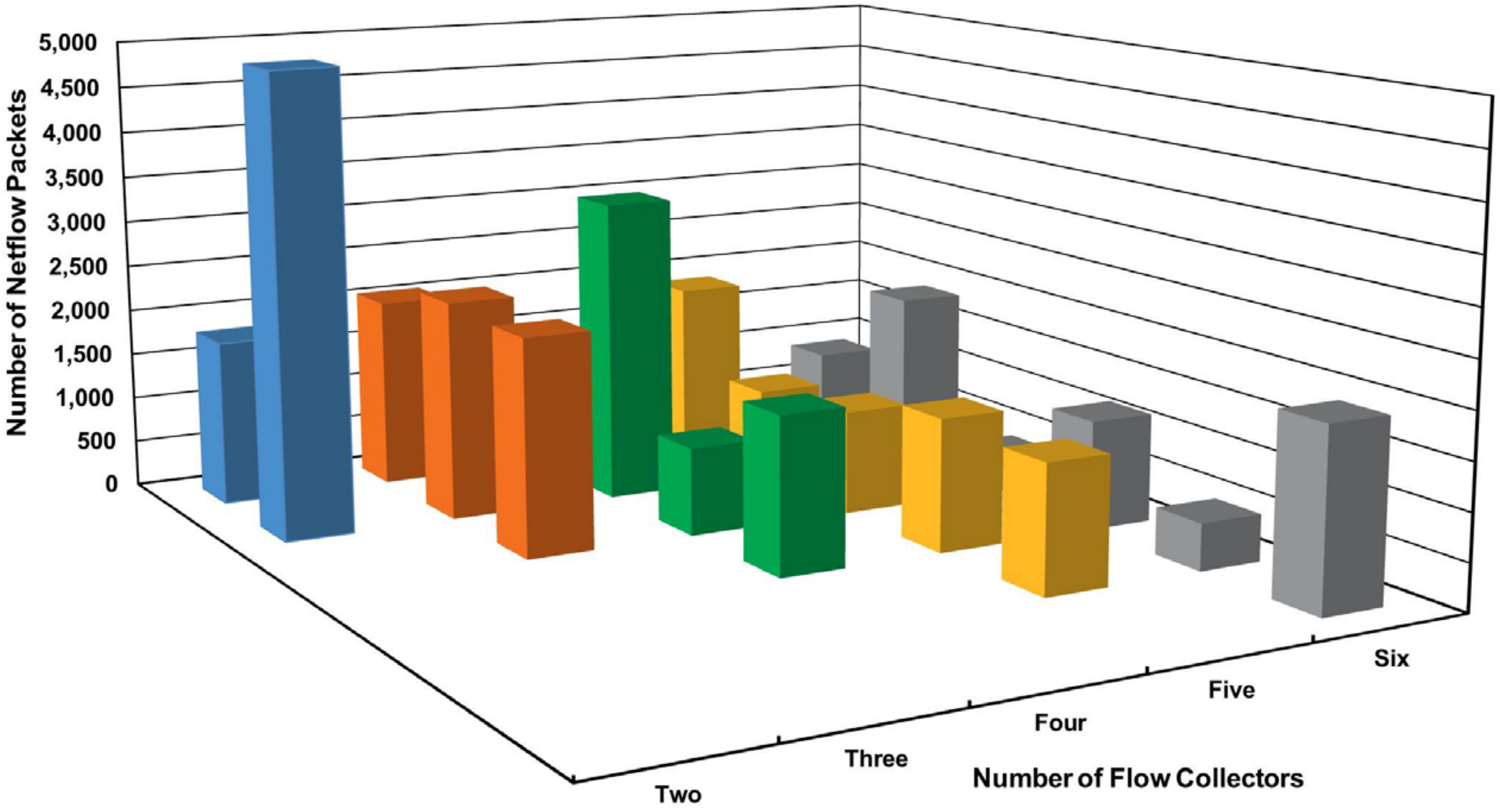
(‘flowd’ collector achieved 635 kfps in one configuration, and UDP Netflow Router clocked 2.5 Mfps)

Load Balancing Multiple Netflow Collectors

- Network flow data from a router must always be processed by the same netflow collector
  - Netflow V9 and IPFIX devices periodically send templates, which the collector uses to parse data records from that device
  - Therefore, we must load balance based on packet counts
- The only information for distinguishing Netflow streams is the source IP address and Netflow version
  - Parsing the contents of the Netflow packet is flowcap's job
- Hashing source IP address based on an even number of split streams yields unsatisfactory load balancing
- Hashing source IP address based on odd or prime number of flowcap instances yields satisfactory load balance

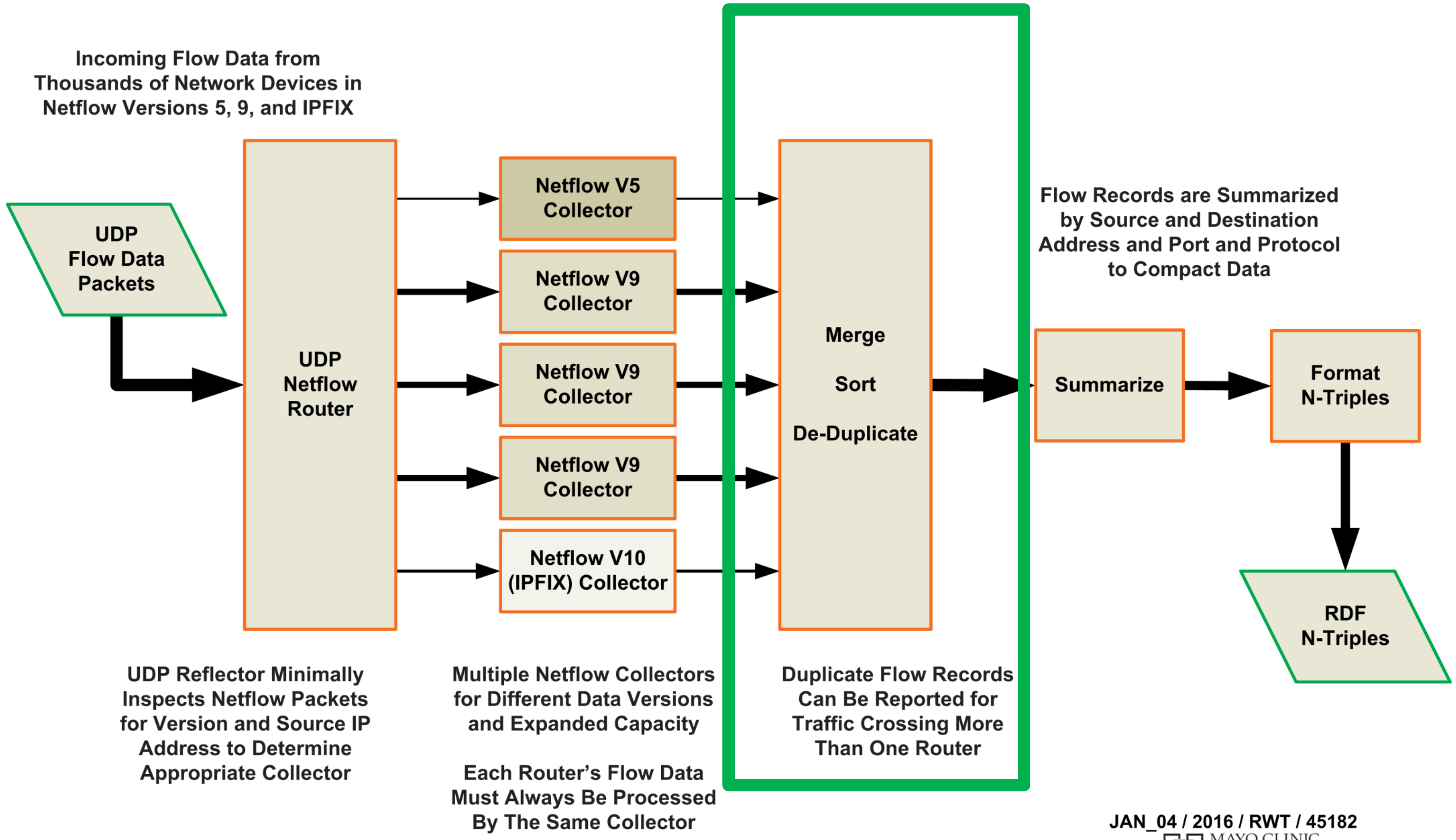
DISTRIBUTION OF NETFLOW PACKETS FOR LOAD BALANCING NETFLOW COLLECTORS USING MODULUS OF ROUTER IP ADDRESS

(Data Sample from 469 Network Routers; Flow Packet Counts Ranged from 1 to 710, Average of 14.4 Packets per Router; Best Load Balancing Achieved with Odd Numbers of Flow Collectors)



JAN_04 / 2016 / RWT / 45180
MAYO CLINIC
SPPDG

SECOND GENERATION NETFLOW EXTRACT-TRANSFORM-LOAD SOFTWARE ARCHITECTURE FOR PARALLEL PROCESSING FLOW RECORDS INTO RESOURCE DESCRIPTION FRAMEWORK (RDF) N-TRIPLE FORMAT SUMMARY DATA


Incoming Flow Data from Thousands of Network Devices in Netflow Versions 5, 9, and IPFIX
UDP Flow Data Packets
UDP Netflow Router
Netflow V5 Collector
Netflow V9 Collector
Netflow V9 Collector
Netflow V9 Collector
Netflow V10 (IPFIX) Collector
Merge
Sort
De-Duplicate
Flow Records are Summarized by Source and Destination Address and Port and Protocol to Compact Data
Summarize
Format N-Triples
RDF N-Triples
UDP Reflector Minimally Inspects Netflow Packets for Version and Source IP Address to Determine Appropriate Collector
Multiple Netflow Collectors for Different Data Versions and Expanded Capacity
Each Router's Flow Data Must Always Be Processed By The Same Collector
Duplicate Flow Records Can Be Reported for Traffic Crossing More Than One Router

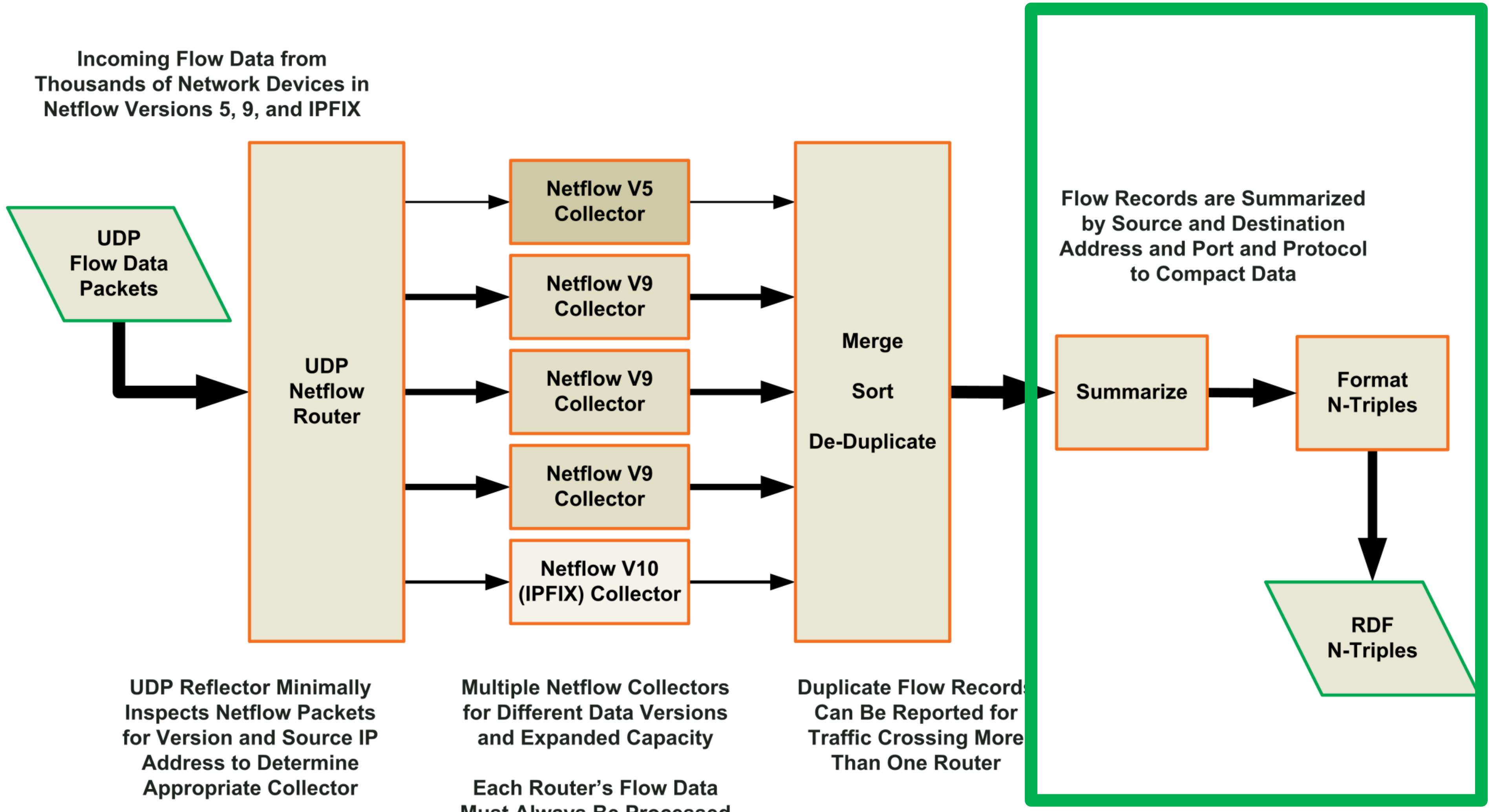
JAN_04 / 2016 / RWT / 45182
MAYO CLINIC
SPPDG

Merging and De-Duplication

- Duplicate records were expected, perhaps coming from different Netflow versions
- SiLK rwdedupe handily performs both functions

- However, rwdedupe performance is problematic
  - First generation C++ program searched only a few hundred sequential records for duplicates
  - rwdedupe must merge and sort all records
  - AND rwdedupe is limited to 4 GB in-memory buffering
    - Compute nodes have 12 cores and 32 GB DRAM
  - De-duplicating 10 minutes of raw data takes 22 minutes

SECOND GENERATION NETFLOW EXTRACT-TRANSFORM-LOAD SOFTWARE ARCHITECTURE FOR PARALLEL PROCESSING FLOW RECORDS INTO RESOURCE DESCRIPTION FRAMEWORK (RDF) N-TRIPLE FORMAT SUMMARY DATA


Incoming Flow Data from Thousands of Network Devices in Netflow Versions 5, 9, and IPFIX
UDP Flow Data Packets
UDP Netflow Router
Netflow V5 Collector
Netflow V9 Collector
Netflow V9 Collector
Netflow V9 Collector
Netflow V10 (IPFIX) Collector
Merge
Sort
De-Duplicate
Flow Records are Summarized by Source and Destination Address and Port and Protocol to Compact Data
Summarize
Format N-Triples
RDF N-Triples
UDP Reflector Minimally Inspects Netflow Packets for Version and Source IP Address to Determine Appropriate Collector
Multiple Netflow Collectors for Different Data Versions and Expanded Capacity
Each Router's Flow Data Must Always Be Processed By The Same Collector
Duplicate Flow Records Can Be Reported for Traffic Crossing More Than One Router

JAN_04 / 2016 / RWT / 45182
MAYO CLINIC
SPPDG

Netflow Summarization

- Summarizes on 5-tuple (src/dst addr/port, protocol)
- Computes sum of flow records, packets, bytes, duration
- rwtotal or rwuniq to produce ASCII output
  - rwuniq is required for TCP flags and ICMP type and code
- Summarize over time period for one data file
  - For 10 minutes of data (~200 million records)
  - rwtotal takes about 10 minutes
  - rwuniq takes about 16 minutes
  - Average of 5.4 flow records per summary
    (approximately 80% compaction)

Implementation Details: Processes and Files

- Netflow data is “always on”, arriving via 10 GbE
- Control scripts and configuration files manage UDP Netflow routing and multiple instances of flowcap
- SLURM batch queue manages de-duplication and translation processes
- File sizes are determined by selection of 10 minute data slices
  - Uncompressed flowcap output files are limited to 4 GB, and typically total 14 GB (2 TB per day)
  - Compressed de-duplicated files are 1.5 GB (0.2 TB per day)
  - ASCII Summary files are 3.3 GB (0.5 TB per day)
  - N-Triples files are 45-90 GB each (8 TB per day)

Implementation Details: Real Time Processing

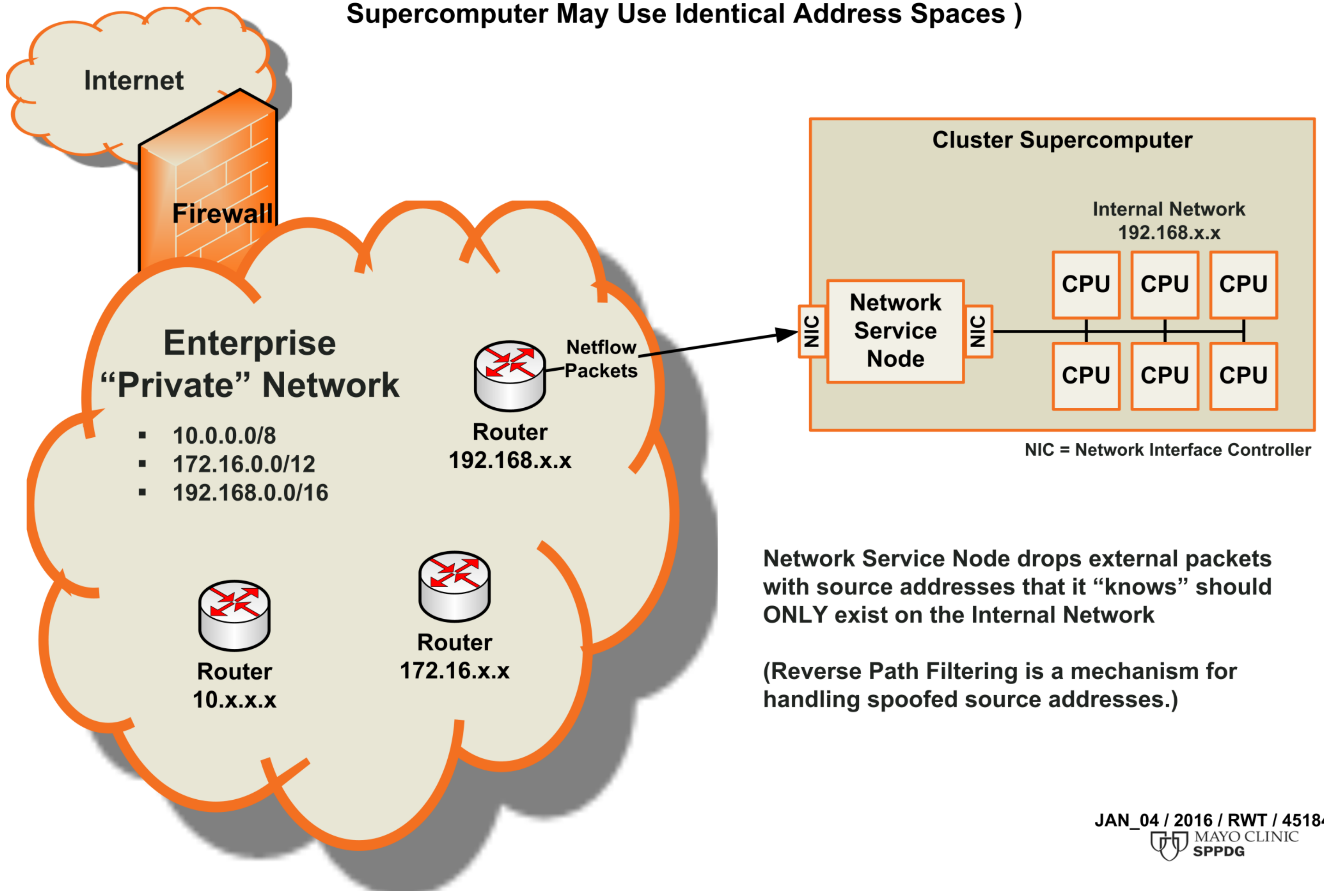
- This pipeline is NOT "real time" or even "near real time"
- Front end UDP Netflow Router and parallel flowcap processes must keep up without packet drops
  - No "stock" instrumentation or monitoring for network interface card (NIC), operating system buffers, or collector software
  - Streaming data record / playback for performance testing is challenging in this particular machine environment
- Post-processing captured flow data is slow
  - De-duplication (22 minutes), summarization (10 minutes), and translation (10 minutes) are batch jobs, resulting in total time to usable RDF data of 42 minutes

Implementation Details: Reverse Path Filtering

- Netflow processing depends on IP packet source address
  - Identifies router or network device providing the report
  - Required to match Flow Templates with Flow Data (V9 and IPFIX)
- However, there can be address conflicts
  - Enterprise network utilizes RFC 1918 private address spaces
    - 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
    - Netflow reports can originate from any of these addresses
  - Computer clusters, clouds, or supercomputers may have their own private address space for the internal network (e.g., 192.168.0.0/16)
- Cluster network interface nodes, with a Linux kernel and IP stack, may discard external packets from addresses matching the internal network

NETFLOW DATA CAPTURE AND PROCESSING ON A SUPERCOMPUTER CAN BE COMPLICATED BY REVERSE PATH FILTERING OF PRIVATE IP NETWORK ADDRESS SPACE

(RFC 1918 Defines IPV4 Private Network Address Spaces; Both Enterprise Network and Supercomputer May Use Identical Address Spaces)



Network Service Node drops external packets with source addresses that it "knows" should ONLY exist on the Internal Network

(Reverse Path Filtering is a mechanism for handling spoofed source addresses.)

JAN_04 / 2016 / RWT / 45184
MAYO CLINIC
SPPDG

Potential Improvements

- Construct a robust test framework
  - Capture or synthesize Netflow export packets (in several versions)
  - Play back at varying speeds, with peak bursts
  - Instrument hardware, OS, and software pipeline to identify performance bottlenecks
- Remove 4 GB memory buffer limit from SiLK tools
- Add millisecond resolution to rwuniq
- Explore optimizations within the SiLK toolset, such as pre-sorting inputs to rwdedupe
- Replace general purpose SiLK tools with custom applications, ported from the first generation pipeline

Summary

- It is possible to capture and process large and complex flow data streams using the SiLK tool set
  - Hundreds of thousands of flows per second
  - Thousands of reporting devices
  - Multiple flow data versions
- Any implementation may face similar issues
  - Splitting the flow data into multiple streams for parallel processing
    - And dealing with the idiosyncrasies of the parallel environment
  - Aggregating parallel data files
  - Summarizing and formatting for analysis (optional?)