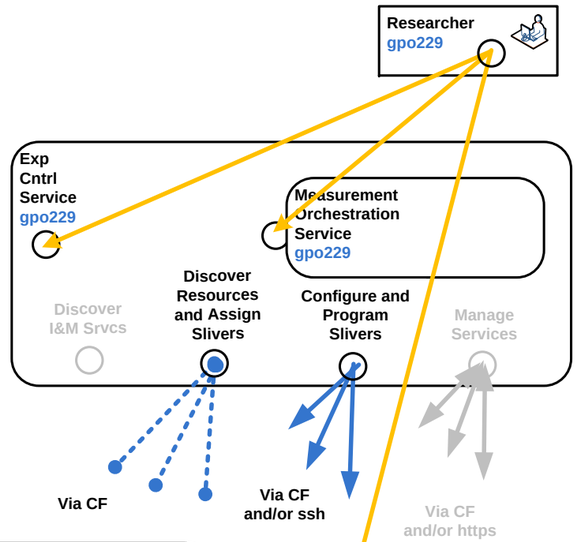
Instrumentation Tools for Researchers (Students)

* Each instance contained in one Aggregate (site)
* Integrated with protoGENI CF
* Meas Orch (MO) srvc:
  * Adds MO srvc to protoGENI Exp Cntrl Srvc that runs in Researcher's browser
  * When Res specifies RSpec for experiment slice, it adds host for *Meas Controller* (MC) sliver
  * Uses info from manifest to identify slivers to be monitored, and dynamically creates config files for SNMP daemon and other capture software
  * Then, it downloads software for *MC* sliver, and it adds MP srvc to each *Experiment Sliver*



Manage Srvcs

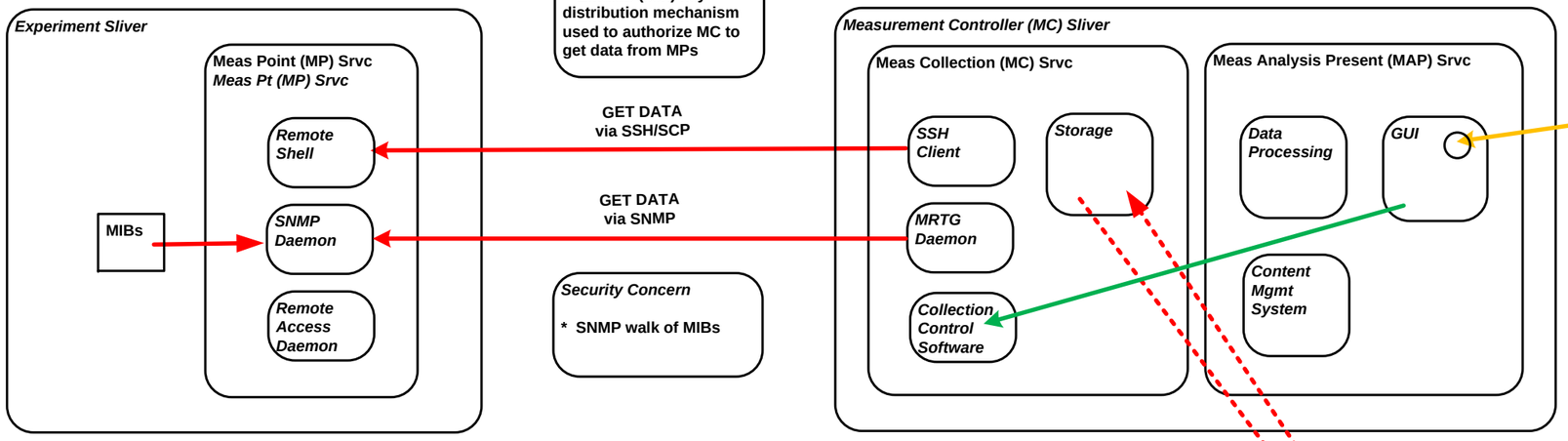
* *MC Srvc* has collection control software
* *MP Srvc* includes remote access daemon to execute capture software
* How?

MP Srvc is automatically loaded into each Experiment Sliver

MC Sliver is automatically loaded into each Experiment Slice

Authorization

* Emulab (ssh) key distribution mechanism used to authorize MC to get data from MPs



Security Concern

* SNMP walk of MIBs

Portal to MCs

* How does this work?

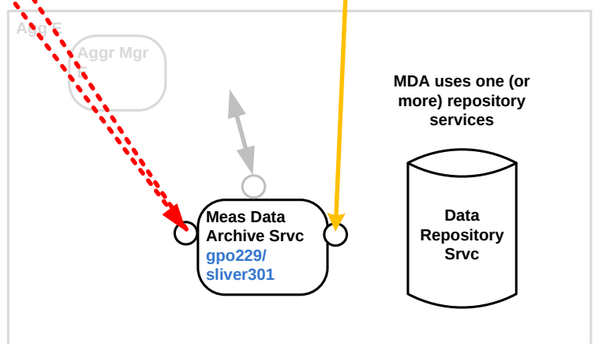
Software	Information	Display Type
SNMPd	Routing Table	Table
	IP Traffic	Graph
	ICMP Traffic	Graph
	TCP Traffic	Graph
	UDP Traffic	Graph
	CPU Utilization	Graph
	Memory Utilization	Graph
	Total Network Traffic	Graph
	Link-specific Traffic	Graph
	Link-specific Unicast Traffic	Graph
ssh/arp	ARP cache	Table
ssh/netstat	TCP streams	Table
ssh/netstat	UDP listeners	Table
ssh/ps	Process list	Table
ssh/lsmod	Installed Kernel Modules	Table

***MP Srvc* includes:**

* SNMP daemon, using existing MIBs, or added MIBs
* tcpdump
* netflow
* custom monitoring code based on pcap library to collect packet stats not captured by SNMP daemon
* ps, vmstat, and SNMP, to capture OS info, such as CPU load, memory load, routing table configs, ARP caches, loaded modules, etc.
* remote access daemon to to execute capture software

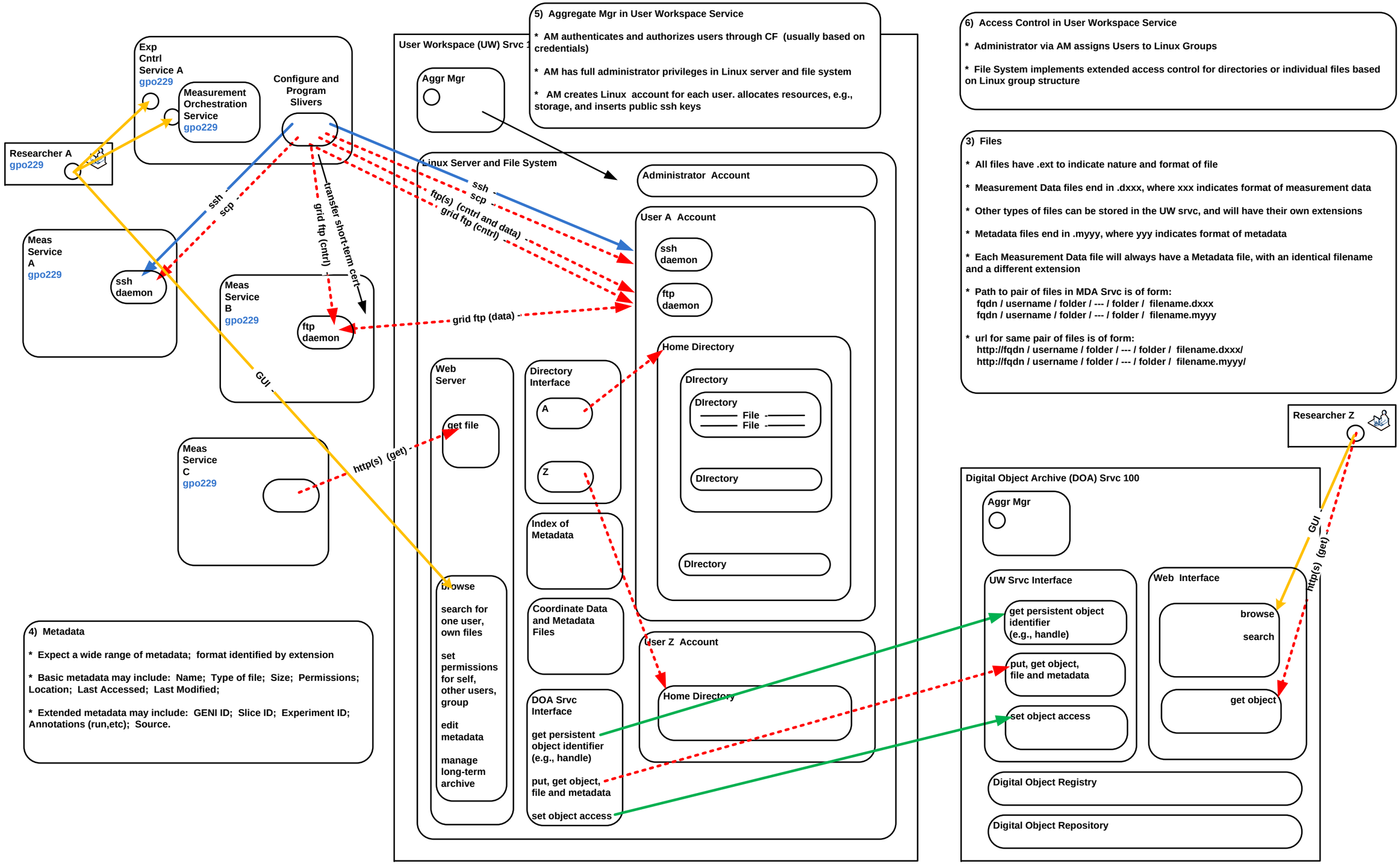
Meas Data Schema

* none defined
* metadata not defined



Meas Data Arch Srvc

* srvc not yet defined
* messages not yet defined


Exp Cntrl Service A gpo229
Measurement Orchestration Service gpo229
Configure and Program Slivers
Researcher A gpo229
User Workspace (UW) Srvc 1
Aggr Mgr
5) Aggregate Mgr in User Workspace Service
* AM authenticates and authorizes users through CF (usually based on credentials)
* AM has full administrator privileges in Linux server and file system
* AM creates Linux account for each user. allocates resources, e.g., storage, and inserts public ssh keys
Linux Server and File System
Administrator Account
User A Account
ssh daemon
ftp daemon
Home Directory
DIrectory
DIrectory
File
File
DIrectory
DIrectory
Meas Service A gpo229
ssh daemon
Meas Service B gpo229
ftp daemon
Meas Service C gpo229
ssh
scp
transfer short-term cert
grid ftp (cntrl)
ftp(s) (cntrl and data)
grid ftp (cntrl)
grid ftp (data)
GUI
http(s) (get)
Web Server
get file
browse
search for one user, own files
set permissions for self, other users, group
edit metadata
manage long-term archive
Directory Interface
A
Z
Index of Metadata
Coordinate Data and Metadata Files
DOA Srvc Interface
get persistent object identifier (e.g., handle)
put, get object, file and metadata
set object access
User Z Account
Home Directory
6) Access Control in User Workspace Service
* Administrator via AM assigns Users to Linux Groups
* File System implements extended access control for directories or individual files based on Linux group structure
3) Files
* All files have .ext to indicate nature and format of file
* Measurement Data files end in .dxxx, where xxx indicates format of measurement data
* Other types of files can be stored in the UW srvc, and will have their own extensions
* Metadata files end in .myyy, where yyy indicates format of metadata
* Each Measurement Data file will always have a Metadata file, with an identical filename and a different extension
* Path to pair of files in MDA Srvc is of form:
fqdn / username / folder / --- / folder / filename.dxxx
fqdn / username / folder / --- / folder / filename.myyy
* url for same pair of files is of form:
http://fqdn / username / folder / --- / folder / filename.dxxx/
http://fqdn / username / folder / --- / folder / filename.myyy/
Researcher Z
Digital Object Archive (DOA) Srvc 100
Aggr Mgr
UW Srvc Interface
get persistent object identifier (e.g., handle)
put, get object, file and metadata
set object access
Web Interface
browse
search
get object
Digital Object Registry
Digital Object Repository
4) Metadata
* Expect a wide range of metadata; format identified by extension
* Basic metadata may include: Name; Type of file; Size; Permissions; Location; Last Accessed; Last Modified;
* Extended metadata may include: GENI ID; Slice ID; Experiment ID; Annotations (run,etc); Source.

perfSONAR – PS
services written in perl
follow SOA web services approach
run on Linux

Meas Info (MI) Srvc

Two (or more) gLSs.

Receive REGISTERs only from hLSs

Contain for each hLS: name, location, metadata summary of available srvcs

Update each other

Global Lookup Srvc (gLS)

UPDATE

Global Lookup Srvc (gLS)

Topology Srvc (TS)

QUERY, ADD, UPDATE, REPLACE

TS Query Request
TS Add Request
TS Update Request
TS Replace Request

One hLS for each entity/domain/aggregate

Receives REGISTERs from MA Srvcs, including: name, location, metadata of available srvcs

Takes registered info and compiles metadata summary of available srvcs

Each hLS registers with one (or more) gLSs, providing name, location, metadata summary of available srvcs

Home Lookup Srvc (hLS)

REGISTER
LS Register Request
LS Deregister Request
LS Keepalive Request

QUERY (first)
LS Query Request - XQuery
LS Query Request – Discovery
LS Key Request

Operators setup long-running I&M services and register them, so that they can be discovered and used by Operators and (when permitted) Researchers

Each MA srvc registers with a hLS, providing name, location, metadata of available srvcs

REGISTER
LS Register Request
LS Deregister Request
LS Keepalive Request

QUERY (second)
LS Query Request - XQuery
LS Query Request – Discovery
LS Key Request

Manage Services GUI

* perfAdmin (CGI script to locate and manage perfSONAR services and data)

* How does this work?

Switch B1
LAN
MIBs

Meas Point (MP) Srvc

Meas Pt (MP) Srvc

Meas Archive (MA) Srvc

Meas Analysis Present (MAP) Srvc

Client

GUI

Operator
ops003

REQUEST (third), SETUP DATA

Echo Request
Metadata Key Request
Setup Data Request

***MP Srvc* types**

* command line
* status via SNMP (e.g., interface counter)
* status via TL1 (e.g., optical devices)
* telnet/SSH
* perfSONAR-BUOY, OWAMP (one-way latency)
* perfSONAT-BUOY, BWCTL (bandwidth testing)
* PingER, Ping scheduled testing (round trip latency)

***MA Srvc* types**

* RRD database
* SQL database

All perfSONAR Messages
addressed to each service at a service URL
formatted in XML using SOAP over HTTP
always a Request and then a Response
(currently) no encryption
(currently) no authentication and authorization
(since Authentitcation Srvc (AS) not yet built or deployed)

Each message follows perfSONAR schema, and contains;
message container
one or more one metadata elements
zero or more data elements

***Client* types**

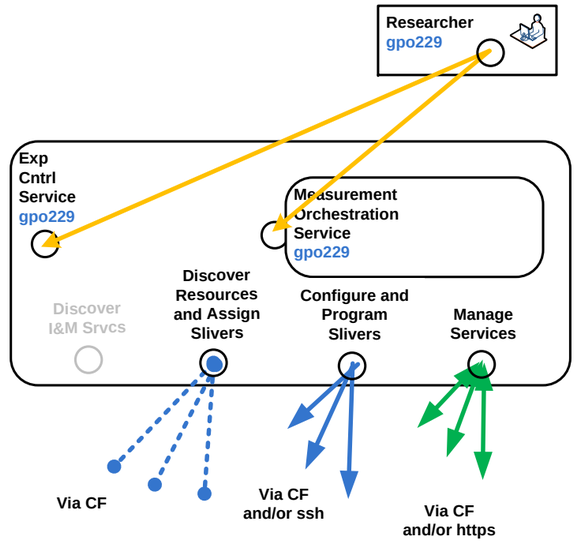
* match *MP Srvc*s
* command line
* status via SNMP and TL1
* telnet/SSH
* perfSONAR-BUOY, OWAMP and BWCTL
* PingER

***GUI* types**

* active Service
* GMAPS
* acad (Java-based visualization)
* E2EMon (link monitoring)
* ESNet (domain utilization)
* trace (traceroute visualization)
* perfAdmin (CGI script to locate and manage perfSONAR services and data)
* perfER GUI (displys the results of pingER testing)
* perfSONAR-BUOY (displays the results of latency and bandwidth testing)

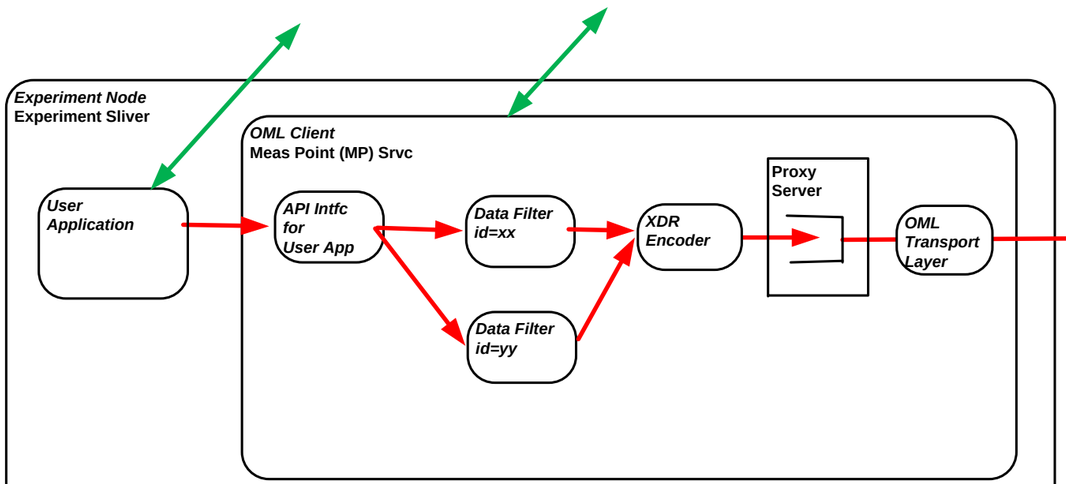
OMF/OML Structure

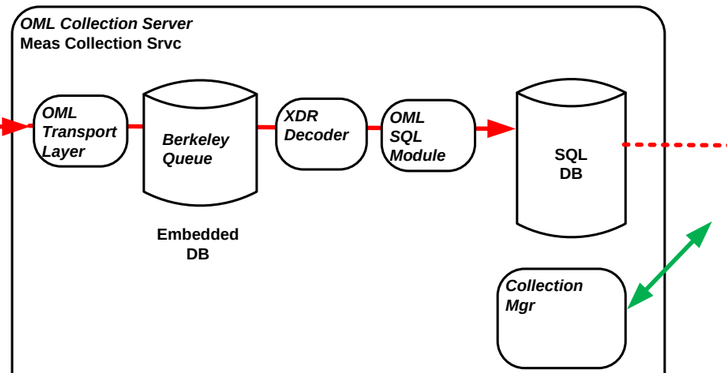
* One *OML Client* included in each *Experiment Node*
* One *OML Collection Server* provided for each OMF site
* Researcher programs *User Application* and *OML Client* to gather and filter desired measurement data, and define stream to Collection Server
* Meas Data streamed from *OML Client* to *OML Collection Server*

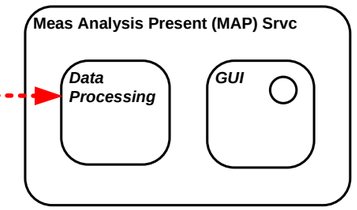


Manage (and Control) Srvcs

* Via HTTP to all srvc's, with APIs based on REST.
* Via HTTP to OML Client srvc, to config files specifying filtering and streaming, which are then compiled into code







Meas Analysis Present Srvc

* Running outside of OMF/OML.
* Can import directly from SQL DB
* EC can arrange to convert tables into graphs

Experiment Portal

* Early prototype
* Each experiment results in a separate page containing all the experiment related information (script, parameter, resources used, time) as well as a pointer to the measurement database.
* Where?

Client API

* Researcher uses web interface to define measurement points and parameters through a web interface, saved as XML config file; causes XSLT-based code generator to generate source code for measurement client.
* Researcher includes measurement-point id's, and metric id's, which dynamically creates a schema.

Meas Data Filters

* Researcher defines filters, separate from User App, so that can easily change data collection behavior
* Can be triggered by various properties, e.g., time, by the no of data values collected, etc.
* Can be triggered by events, e.g., passing a trip line in a mobile application

Meas Data Streams

* Researcher defines measurement streams, gathering data samples and averaging, etc.
* Meas data is series of typed vectors, XDR coded, and then streamed from client to collection server using proprietary OML protocol, on top of TCP, over dedicated Control VLAN
* Considering using IPFIX instead of prop OML protocol; IPFXI typically uses SCTP for transport
* If path becomes disconnected from time-to-time. data is cached in Proxy Server FIFO, and then forwarded when path is reestablished

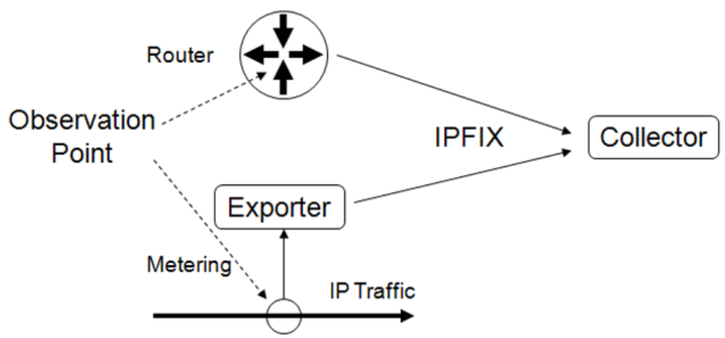
Meas Data Schema

* Meas data follows schema defined by researcher, including: measurement-point id's, metric id's, etc.
* A sensor (or application, or service) define a set of measurement points, with each measurement point defined by a name and a typed vector (sensor schema).
* At runtime, the experimenter (or operator) provides a streams spec which defines what measurement points are going to be activated and what initial processing is going to be performed - that defines the actual schema going over the wire and/or ending up in the collection database

DB Schema

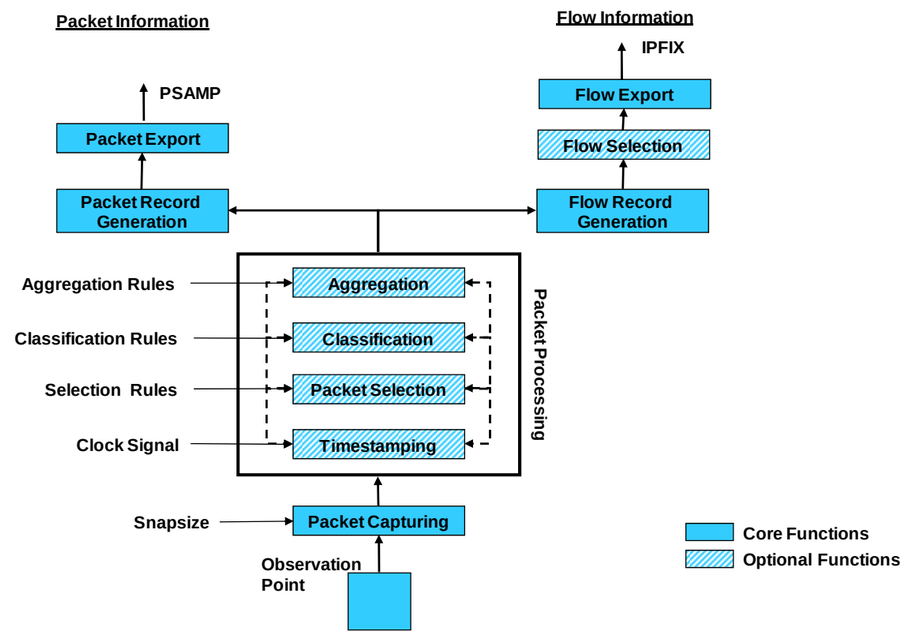
* Application definition is used to create DB schema for experiment, using XSLT.
* DB table is created for each measurement point, names based on id attribute of the group element.
* Includes mandatory fields for name/ id, timestamp, sequence number
* Protocol is self describing
* Server automatically creates a table for every distinct stream (distinct in terms of schema not source).
* Streams carry their own name which is translated into a database using a simple naming convention.

IPFIX Architecture



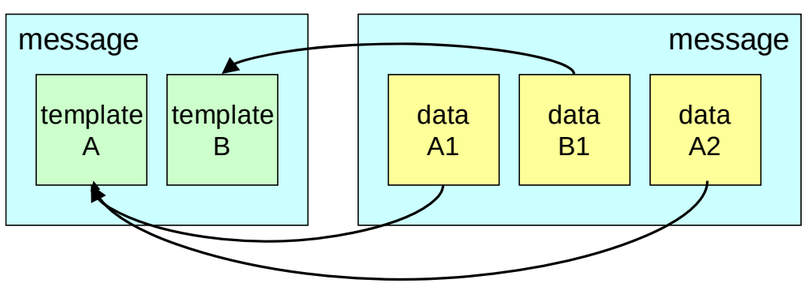
- push protocol: periodically IPFIX messages to configured receivers
- Transport protocols: SCTP (, UDP, TCP)

IPFIX/PSAMP Measurement Model



Data Representation

- Templates in the message stream describe the data sets
- Allows flexible and efficient (binary) representation of flows on the wire



Information Model

- The information model supports reporting a wide variety of information elements (IEs):
  - "Five-tuple" (IPv4, IPv6 header fields) and standard packet/byte counters
  - All ICMP, TCP, UDP header fields
  - Layer 2, VLAN, MPLS, and other sub-IP information
  - Timestamps down to nanosecond resolution
  - Packet treatment: e.g., routed next hop and AS
  - Detailed counters: e.g., sum of squares, flag counters
- New IEs registered with IANA
- Enterprise-specific IEs for private extensions
- New defined IEs
  - location / GPS information, QoS parameters, spectrum measurements, ...

Measurement Data Schema

*** Radar data follows NetCDF format.**

*** NetCDF is: "self-describing, machine independent, binary, data format, that supports the creation, access and sharing of array-oriented scientific data."**

*** "Self-describing" means that there is a header that describes the rest of the file, in particular the data arrays, as well as arbitrary file metadata in the form of name/value attributes.**

*** Various users of NetCDF (such as Climate and Forecast (CF)) have guidelines for metadata.**

References:

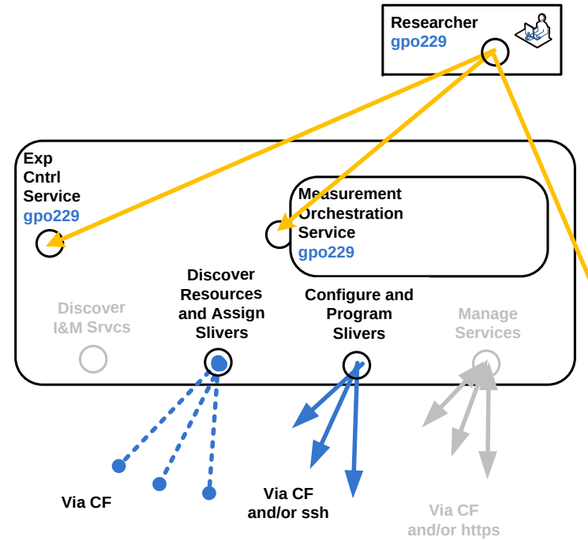
*** NetCDF (network Common Data Form) is a set of software libraries and machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data.**

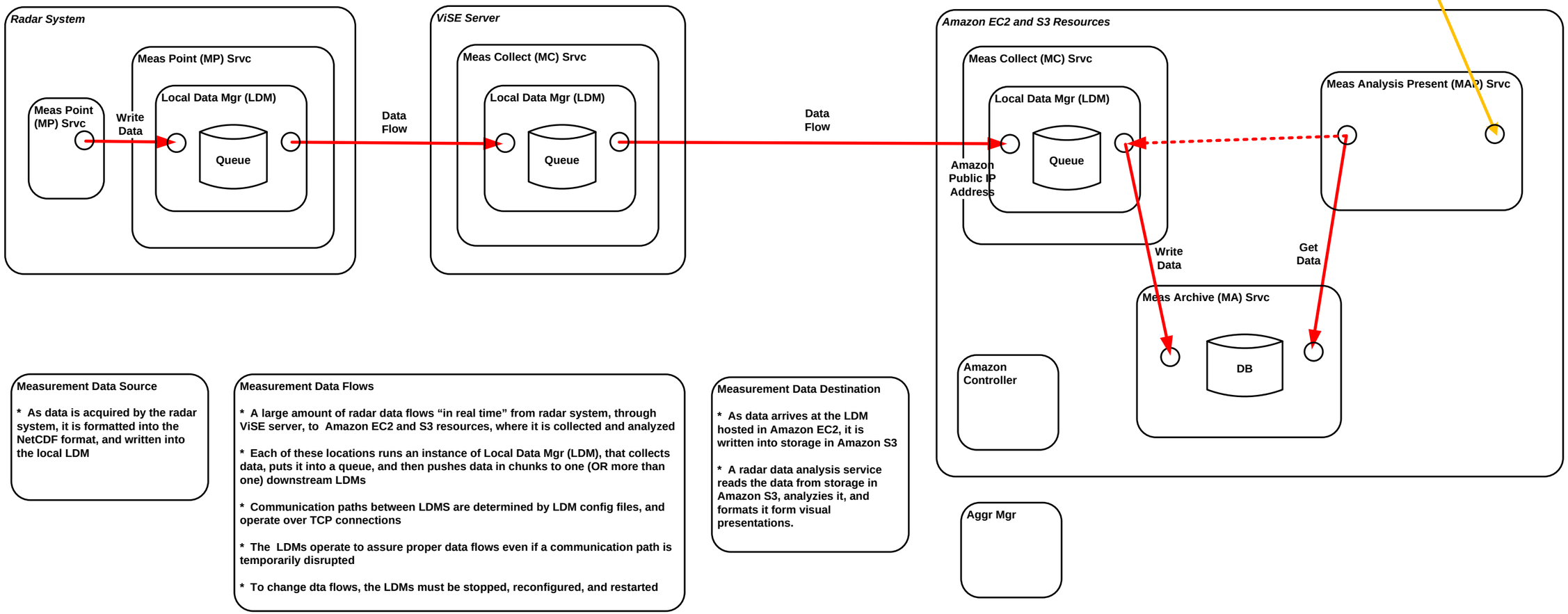
http://www.unidata.ucar.edu/software/netcdf/

*** Unidata Local Data Manager (LDM) is a collection of cooperating programs that select, capture, manage, and distribute arbitrary data products. The system is designed for event-driven data distribution, and is currently used in the Unidata Internet Data Distribution (IDD) project. The LDM system includes network client and server programs and their shared protocols. An important characteristic of the LDM is its support for flexible, site-specific configuration.**

http://www.unidata.ucar.edu/software/ldm/

http://www.unidata.ucar.edu/software/ldm/ldm-6.6.5/basics/generic-LDM.html





Measurement Data Source

*** As data is acquired by the radar system, it is formatted into the NetCDF format, and written into the local LDM**

Measurement Data Flows

*** A large amount of radar data flows "in real time" from radar system, through ViSE server, to Amazon EC2 and S3 resources, where it is collected and analyzed**

*** Each of these locations runs an instance of Local Data Mgr (LDM), that collects data, puts it into a queue, and then pushes data in chunks to one (OR more than one) downstream LDMs**

*** Communication paths between LDMS are determined by LDM config files, and operate over TCP connections**

*** The LDMs operate to assure proper data flows even if a communication path is temporarily disrupted**

*** To change dta flows, the LDMs must be stopped, reconfigured, and restarted**

Measurement Data Destination

*** As data arrives at the LDM hosted in Amazon EC2, it is written into storage in Amazon S3**

*** A radar data analysis service reads the data from storage in Amazon S3, analyzies it, and formats it form visual presentations.**