ANSE and PhEDEx
SDN Demonstration at FTW

Integrating Network-Awareness and Network-Management into PhEDEx

Presented by Azher Mughal (Caltech)
Main Authors: Vlad Lapadatescu & Tony Wildish

Global Lambda Integrated Facility
Technical Working Group Meeting #23
Spring 2015
Overview

- Advanced Network Services for Experiments
- (Short) PhEDEx intro

- Current development efforts w.r.t. circuits and PhEDEx
  - Where/how it can be integrated
  - Previous results (ISGC 2014)

- Circuit awareness PhEDEx
  - Updated FileDownload agent / ResourceManager

- NSI circuits, issues encountered and proposed solution

- Summary and future plans
A project funded by NSF CC-NIE program

**ANSE** - Advanced Network Services for Experiments

Integrate network awareness into the software stacks of experiments
- PhEDEx for CMS
- Panda for ATLAS

Official starting date Jan 2013
- Build on top of existing services (LHCOPN, LHCONE)

PIs:
- Harvey Newman, PI, Caltech
- Shawn McKee, co-PI, University of Michigan
- Paul Sheldon, co-PI, Vanderbilt University
- Kaushik De, co-PI, University of Texas at Arlington
PhEDEx Overview

The data management transfer tool for CMS (since 2004)
Loosely coupled set of agents written in Perl interacting via central DB

- **central agents** (ex. FileRouter agent)
- **site agents** running at various T1s and T2s (ex. FileDownload agent)
- each agent performs a independent single task

PhEDEx front-end and data-service

**Three instances** running over the same network

Common workflow:

- Front-end used to request data to sites
- Central agents compute paths of least cost, schedule transfers, etc
- Site agents execute transfer tasks

FileRouter (central) agent builds transfer queue per destination
FileDownload (site) agent examines its queue, processes it & reports back
PhEDEx is:
• not necessarily “near” the storage (i.e. same subnet)
• high level software ... only knows about:
  • datasets, blocks, files
  • Hostnames/IPs from URLs
  • Path of files

When issuing a transfer request user supplies:
• Dataset/block
• Destination site(s)

Data that PhEDEx can provide
• Datasets, blocks & file names & sizes
• SURL (storage farm hostname, local file path)
• Information about transfer queues
• Limited monitoring information
PhEDEx transfers over the past 5 years

CMS PhEDEx transfers

- All transfers
- T1->T2 transfers
- T2->T2 transfers

- 2010-2011
- 2011-2012
- 2012-2013
- 2013-2014 (LS1)
- 2014-2015 (LS1)
ANSE & PhEDEx

Goals:
• Enhance PhEDEx with circuit awareness capabilities
• Provide a tool which can be used by others

Motivation*:
• More deterministic transfers (schedule jobs with data)
• Data prioritization over other traffic

PhEDEx integration possibilities:
• In the FileDownload agent (site level):
  + Compromise between desired functionality and complexity
  - Only has a local view

• In the FileRouter agent (central level):
  + Has a global view of the whole system
  - Harder to implement and optimize

* Provided that guaranteed BW is available
Initial prototype

Modified the FileDownload agent to:

- Check its own download queue
- Determine whether a circuit is needed
- Request a circuit (using DYNES)
  - If circuit was established:
    - convert transfer URLs to use the new L3 path
    - start new transfer using the updated URLs
- Manage the lifecycle of the circuit

Prototype:

- Required no modifications to PhEDEx DB
- Had all control logic in the FileDownload agent
- Was transparent for all other PhEDEx instances

Issues:

- Relied on FDT as a transfer backend
- Could not be used by external apps
- Could not be extended to use other circuit providers
Results using the prototype

- **Seamless path switch**
- **Per job link rates with PhEDEx traffic**
  - ~620MB/sec -> 1060 to 1250MB/sec
- **Average link rates with PhEDEx traffic**
  - ~570MB/sec -> ~1050MB/sec

Limited by background traffic
Integrating circuit awareness in PhEDEx – inner workings

Standard FileDownload agent:
• Files from the transfer queue are grouped into transfer jobs
• Jobs are handed to the transfer backend (FDT, FTS, etc...) for execution
• Transfer backend reports back with transfer status
• FileDownload agent reports back to DB

Updated FileDownload agent (CircuitAgent):
• Determines whether a circuit is worthwhile and requests one if it is
• Circuit request goes via the ResourceManager
• When a new transfer job is ready to start
  • Checks if a circuit is available (via ResourceManager)
  • Updates job to use circuit instead of GPN

ResourceManager:
• Interacts with PhEDEx via direct calls
• Interacts with external programs via a REST interface
• Handles the lifecycle of the circuit on behalf of those programs
• Can handle different types of circuits (via plug-ins)
Class diagram

High level circuit management software (can function independent of PhEDEx)

FileDownload agent

CircuitAgent

ResourceManager

HTTPServer

Resource

Circuit

Bandwidth

External

NSI

Dummy

OSCARS* (Dynes)

Backends

Core

IDC

REST calls (JSON)

External agents (PanDA, etc)

1..1

1..1

1..1

Extends

Extend

Extends

Extend

(1..n)

0..n

(1..n)
checkWorkload

shouldRequestCircuit

60s

loop

for all paths which allow & benefit from circuits

requestCircuit

canRequestCircuit

checkLinkSupport

yes/no

ResourceManager

Backends

requestTimeout(requestTimeout)

handleRequestResponse

if

[Circuit request succeeded]

[Circuit request failed]

verifyStateConsistency

Circuit added to the list of available circuits
Path blacklisted for a finite period (usually 1h)

tear down(lifetime)

backendTeardownCircuit

transferTask

isCircuitAvailable

if

[circuit (if available)]
Transfer of file takes place on circuit (circuit_IPs_used)
Transfer of file takes place on default route (TFC_IPs_used)

[circuit exists]

[No circuit]
Using NSI

Network Service Interface
• NSI is an advance-reservation based protocol
• Supports tree and chain model of service chaining

Two phase reservation system
• First phase: availability is checked, if available, resources are held
• Second phase:
  • the requester either commits or aborts a held reservation
  • should the requester fail to do the above, a reservation can expire after a set timeout

NSI reservation properties
• Source, destination endpoints (mandatory)
• Start time, end time, reserved bandwidth (optional)

Limitations
• Only supplies a L2 circuit
• Circuit ends at site border router
• Some providers don’t guarantee BW
Issues in dealing with L2 circuits

Transfer backends can’t directly use the NSI L2 circuit

Establishing L3 path to storage requires:
• Some topology knowledge
• Routing information
• Direct access to the site’s network equipment

PhEDEx is a very high-level software -> Can only provide
• Datasets, blocks & file names and sizes
• SURL (Storage URL)
  • Storage farm hostname
  • Local file path

=> Establishing L3 paths is non trivial
Issues in dealing with FTS and SRM

SURLs to TURLs (FTS & SRM)

- Get source TURL (call `srmPrepareToGet`)
- Get destination TURL (call `srmPrepareToPut`)
- Assuming that the TURL-s are gridftp endpoints, start gridftp copy
- Monitor transfer progress
- Release TURLs
Initial discussions

Technical constraints:
• Only a L2 circuit
• L2 circuit ends in the site’s border router
• Limited feedback in case of errors
• NSI adoption in production is still limited

All solutions of creating a L3 path rely either on
• privileged access on site’s servers/routers
• specialised hardware in place (OF capable)

Our solution must:
• deal with sites serving multiple VOs
• potentially deal with privileged and non privileged files transferred from the same server
• work with the FTS/SRM/gridFTP
• be as un-intrusive into sites operations as possible
1. Request circuit between site A and site B
2. PhEDEx specifies, list of files to be transferred
3. Before transfer, gridFTP checks if the file(s) should go on the circuit
4. If that’s the case set up a TC rule: mark packets of files to go on the circuit
5. Set up a PBR (or use OF) to do the routing of those files afterwards

Issues:
- Relies on modifying or developing plug-ins for the transfer tools
- Relies on having privileged access on servers (for packet marking)
Proposed solution diagram

LHC Site 1
Server Farm 1
Server Farm 2

The Internet

Border Router 1

AutoGOLE

Border Router 2

LHC Site 2
Server Farm 3
Server Farm 4

Server Farm 2
SRM/FTS wrapper**

Server Farm 4
SRM/FTS wrapper**

Request Modify Query Teardown

ResourceManager via NSI Plug-in

OpenFlow controller(s)*

External app PanDA, etc.
PhEDEx

* Ensures correct routing once L2 circuit is up
** Retrieves the IPs of gridFTP servers involved in transfer
1. Request circuit between site A and site B
2. Wrapper gets IPs of all servers involved in the transfer
3. Wrapper passes this information to the OF controller
4. PhEDEx informs the OF controller that a circuit has been established between the two sites
5. OF controller adds routing info in the OF switches that direct all traffic on the subnet to the circuit
Summary & future plans

PhEDEx is set to use circuits when they are available
• No modifications done to PhEDEx DB
• Control logic is in the FileDownload agent
• Lifecycle handled by the ResourceManager
• Transparent for all other PhEDEx instances

ResourceManager can be used as a 3rd party tool

Future plans:
• Solve the issue of how to route data once a circuit is active
• Demonstrate circuit management capabilities between select sites
• Demonstrate improvement while using circuits
FTW Demo

Non-SDN Network/Static PATH

IP Addresses:

- Unesp Switch:
- RNP Switch:
- Caltech Switch: 131.215.207.30
- Michigan Switch:
- I2 FSF:
- AmLight FSF: 190.103.184.134
- Caltech Server1: 131.215.207.24
- Caltech Server2: 131.215.207.25
- Unesp Server:
- RNP Server:
- Michigan Server:
- ODL Server: 131.215.207.57
- Monitoring System:
- Michigan Server:
Thank you!