Deployment & Advanced Regular Testing Strategies

ASTRON perfSONAR training

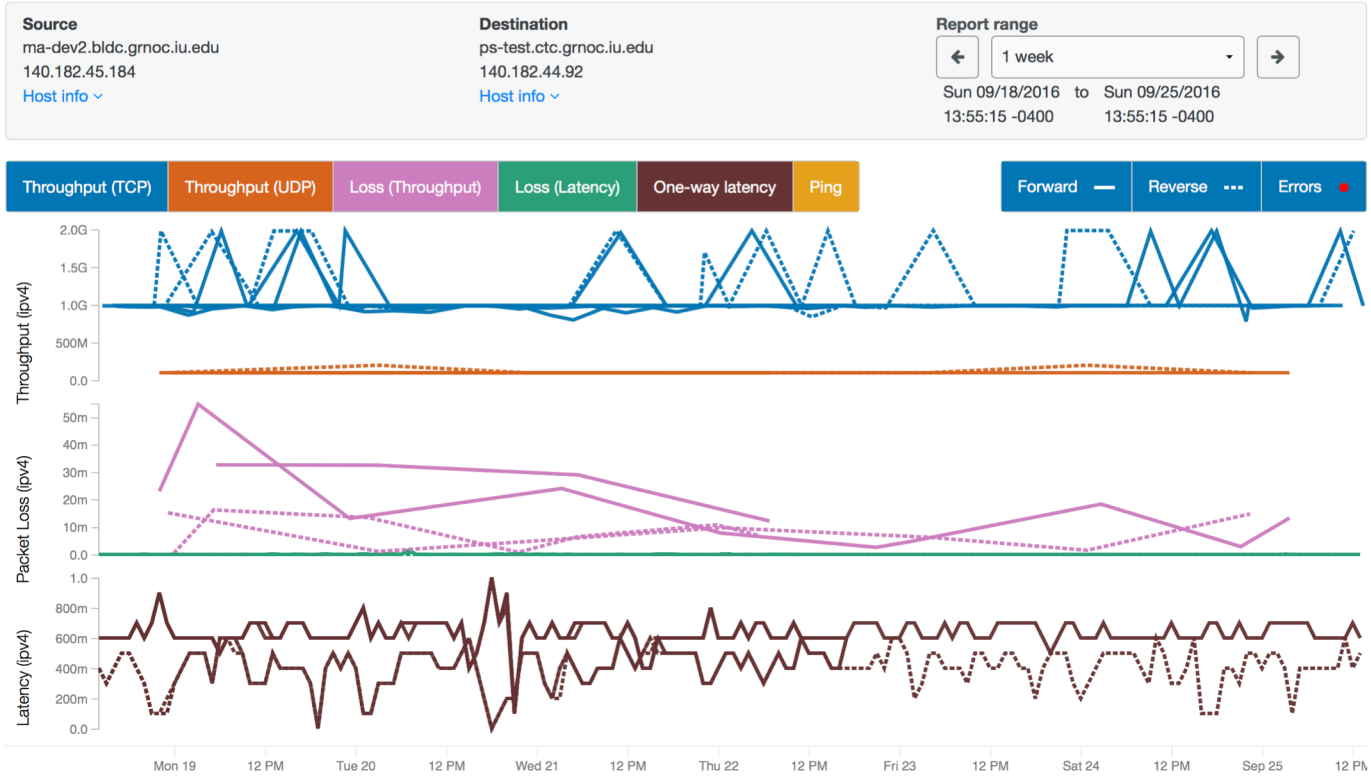
Antoine Delvaux, PSNC, antoine.delvaux@man.poznan.pl
Szymon Trocha, PSNC, szymon.trocha@man.poznan.pl
24-26 September 2018

Importance of Regular Testing

- We can't wait for users to report problems and then fix them (soft failures can go unreported for years!)
- Things just break sometimes
  - Failing optics
  - Somebody messed around in a patch panel and kinked a fiber
  - Hardware goes bad
- Problems that get fixed have a way of coming back
  - System defaults come back after hardware/software upgrades
  - New employees may not know why the previous employee set things up a certain way and back out fixes
- Important to continually collect, archive, and alert on active throughput test results

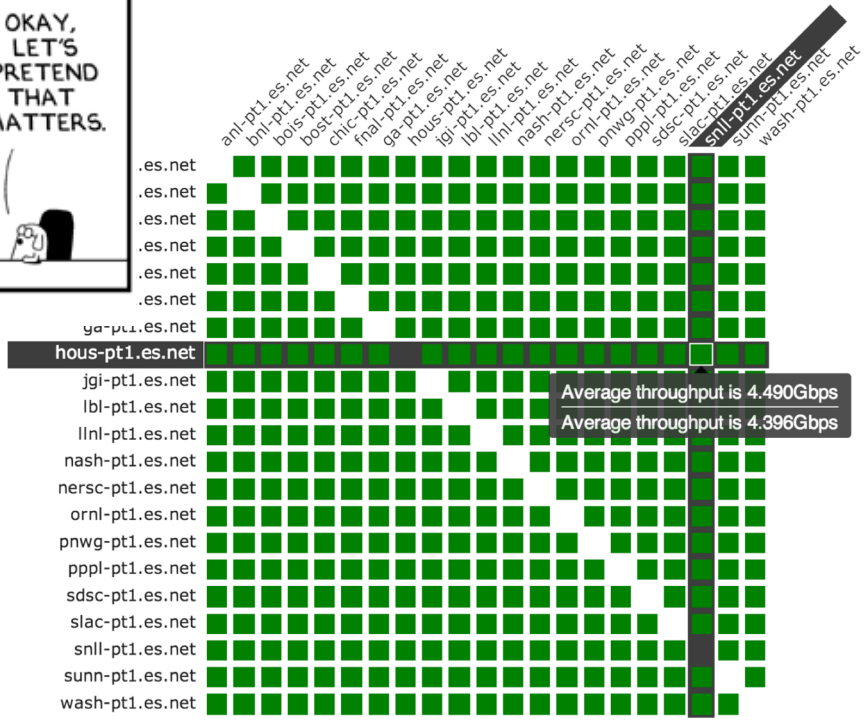
Performance History

perfSONAR



ESnet ENERGY SCIENCES NETWORK GÉANT INDIANA UNIVERSITY INTERNET2 UNIVERSITY OF MICHIGAN

MaDDash: http://ps-dashboard.es.net

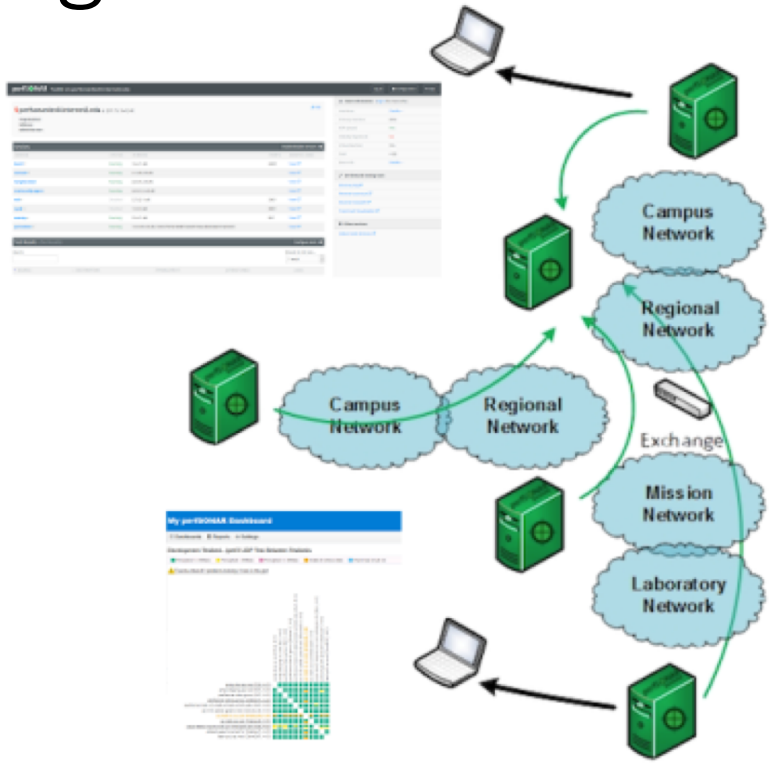


Regular Testing

- There are a couple of ways to do this:
  - Beacon: Let others test to you (e.g. no regular configuration is needed)
  - Island: Pick some hosts to test to – you store the data locally. No coordination with others is needed
  - Mesh: full coordination between you and others (e.g. consume a testing configuration that includes tests to everyone, and incorporate into a visualization)

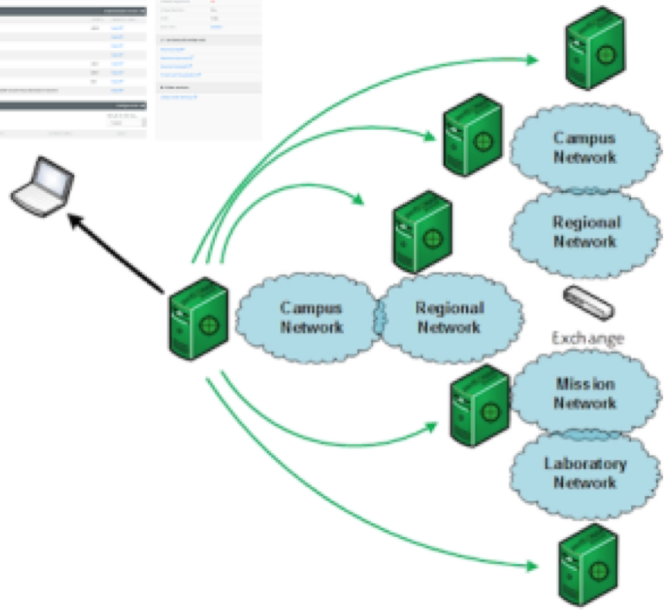
Regular Testing - Beacon

- The beacon setup is typically employed by a network provider (regional, backbone, exchange point)
  - A service to the users (allows people to test into the network)
  - If no regular tests are scheduled, minimum requirements for local storage.
  - Makes the most sense to enable all services (bandwidth and latency)



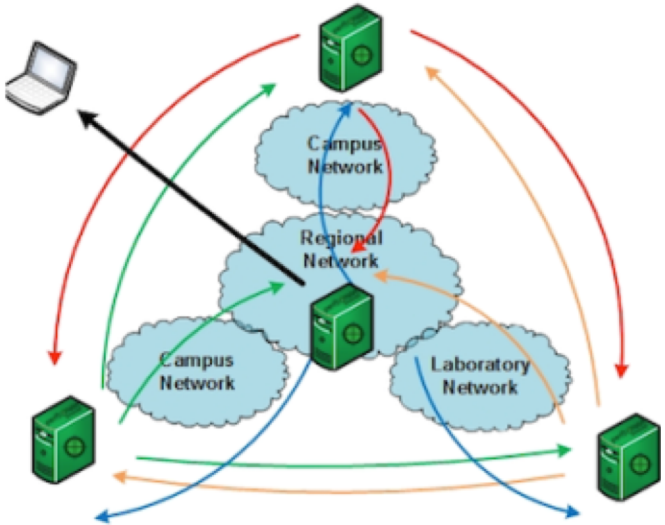
Regular Testing - Island

- The island setup allows a site to test against any number of the 1200+ perfSONAR nodes around the world, and store the data locally.
  - No coordination required with other sites
  - Allows a view of near horizon testing (e.g. short latency – campus, regional) and far horizon (backbone network, remote collaborators).
  - OWAMP is particularly useful for determining packet loss in the previous cases.
  - Throughput will not be as valuable when the latency is small



Regular Testing - Mesh

- A full mesh requires more coordination:
  - A full mesh means all hosts involved are running the same test configuration
  - A partial mesh could mean only a small number of related hosts are running a testing configuration
- In either case – bandwidth and latency will be valuable test cases



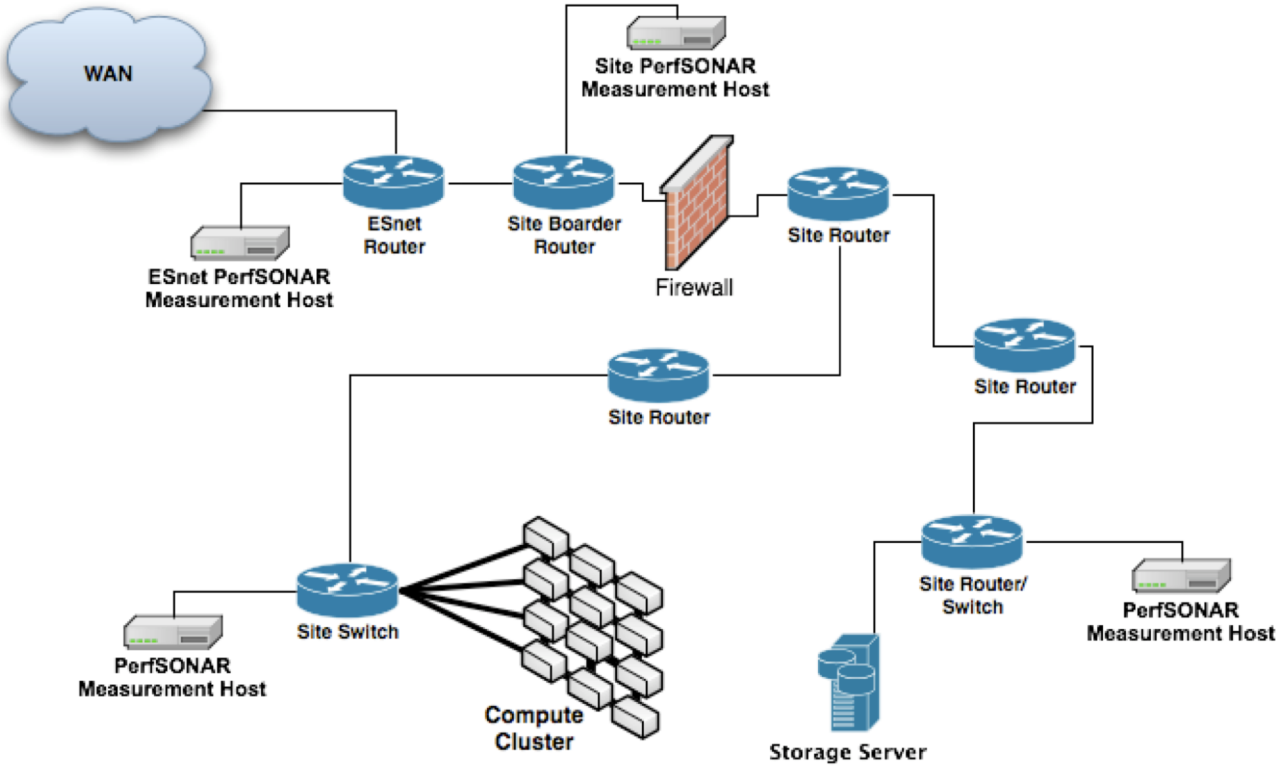
Develop a Test Plan

- What are you going to measure?
  - Achievable bandwidth
    - ***2-3 regional destinations***
    - 4-8 important collaborators
    - 4-8 times per day to each destination
    - 20 second tests within a region, longer across oceans and continents
  - Loss/Availability/Latency
    - OWAMP: ~10-20 collaborators over diverse paths
  - Interface Utilization & Errors (via SNMP)
- What are you going to do with the results?
  - NAGIOS Alerts
  - Reports to user community
  - MaDDash

perfSONAR Deployment Locations

- Critical to deploy near key resources such as DTNs
- More perfSONAR hosts allow segments of the path to be tested separately
  - Reduced visibility for devices between perfSONAR hosts
  - Must rely on counters or other means where perfSONAR can't go
- Effective test methodology derived from protocol behavior
  - TCP suffers much more from packet loss as latency increases
  - TCP is more likely to cause loss as latency increases
  - Testing should leverage this in two ways
    - Design tests so that they are likely to fail if there is a problem
    - Mimic the behavior of production traffic as much as possible
  - Note: don't design your tests to succeed
    - The point is not to "be green" even if there are problems
    - The point is to find problems when they come up so that the problems are fixed quickly

Sample Site Deployment



Managing Multiple Hosts with pSConfig

MaDDash & pSConfig

- Measurement results are more useful when they can be “seen”, because this implies they will be acted on.
- MaDDash is a software package that can be used to visualize the results of many perfSONAR tests
- The pSConfig is a template framework for describing and configuring a **topology** of **tasks**
  - E.g. this is in contrast to the other method of configuration
    - the “Island” model
  - Changes node from ‘testing as an island’ to being a part of a larger testing strategy
- More info: http://docs.perfsonar.net/psconfig_intro.html

pSConfig basic concepts

*A **template** is a description of the task topology in a machine readable format*

- The pSConfig templates are formatted in JSON. The files containing this JSON data are referred to as ***pSConfig templates***.

*A **task** is a job to do consisting of a test to be carried out, scheduling information and other options.*

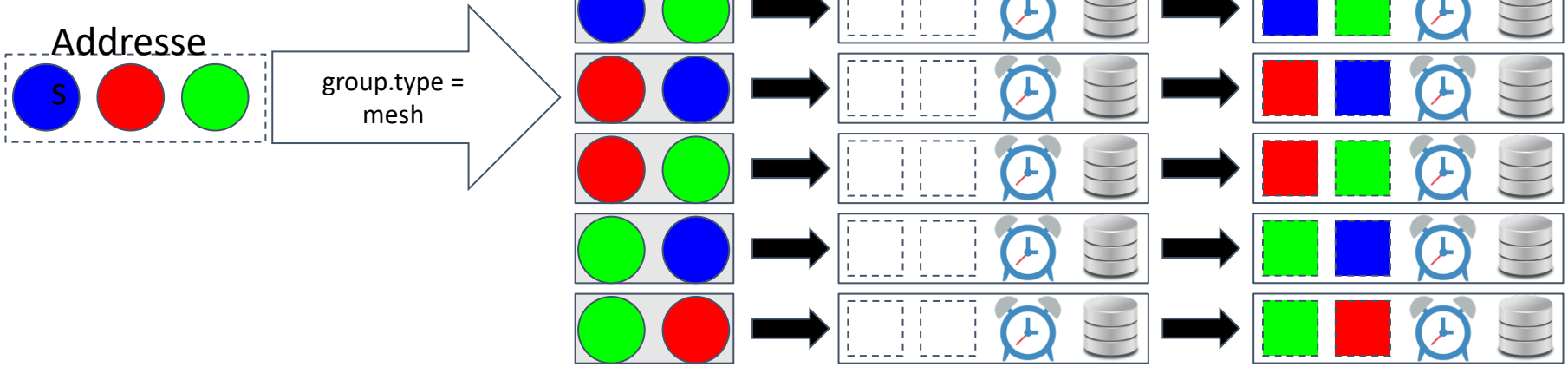
- In pScheduler a single task consists of a number of components, and these elements carry-over to pSConfig: Tests, Tools, Schedules, Archivers, Contexts

*A **topology** is the way in which tasks are interrelated and arranged*

- Ultimately we want a list of tasks to be performed
- Many of these tasks have common components. These common components often represent relationships which we care about when looking at multiple tasks

Creating Tasks

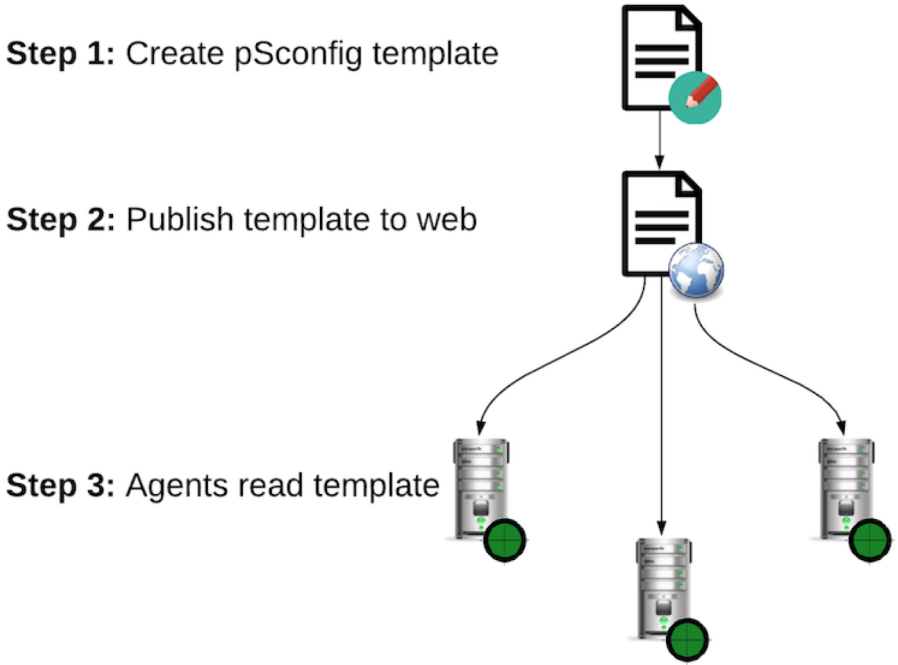
For each pair in the group, we generate a task to be run using properties of the input addresses



Agents

*An **agent** is software that reads one or more pSConfig templates and uses the information to perform a specific function.*

- We currently have two agents:
  - **pscheduler-agent**: It reads the template file(s) and generates pScheduler tasks
  - **maddash-agent:** It reads the the template file(s) and generates a maddash.yaml file



Deployment & Advanced Regular Testing Strategies

ASTRON perfSONAR training

Antoine Delvaux, PSNC, antoine.delvaux@man.poznan.pl
Szymon Trocha, PSNC, szymon.trocha@man.poznan.pl
24-26 September 2018