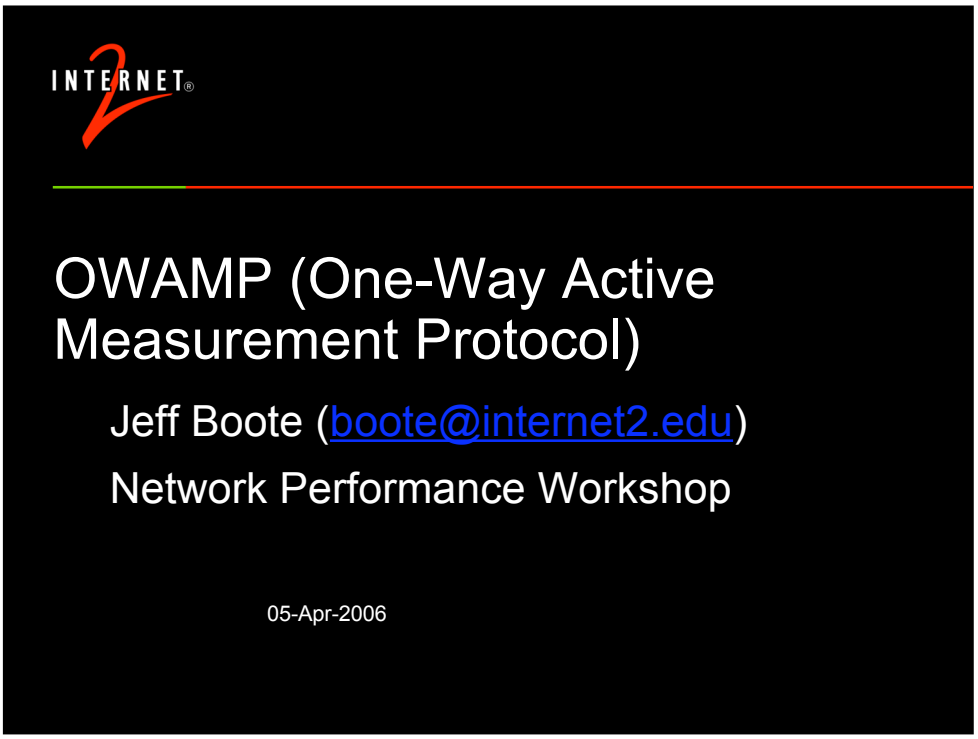
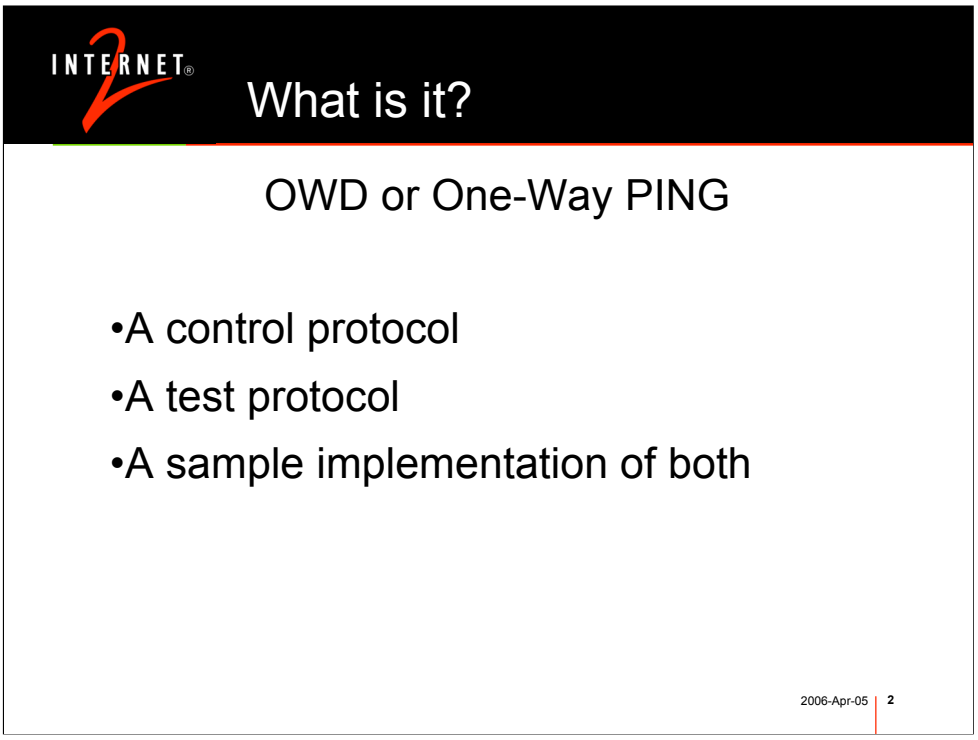


Or… one-way ping



One-Way latencies or Delay. Think of it as one-way ping.

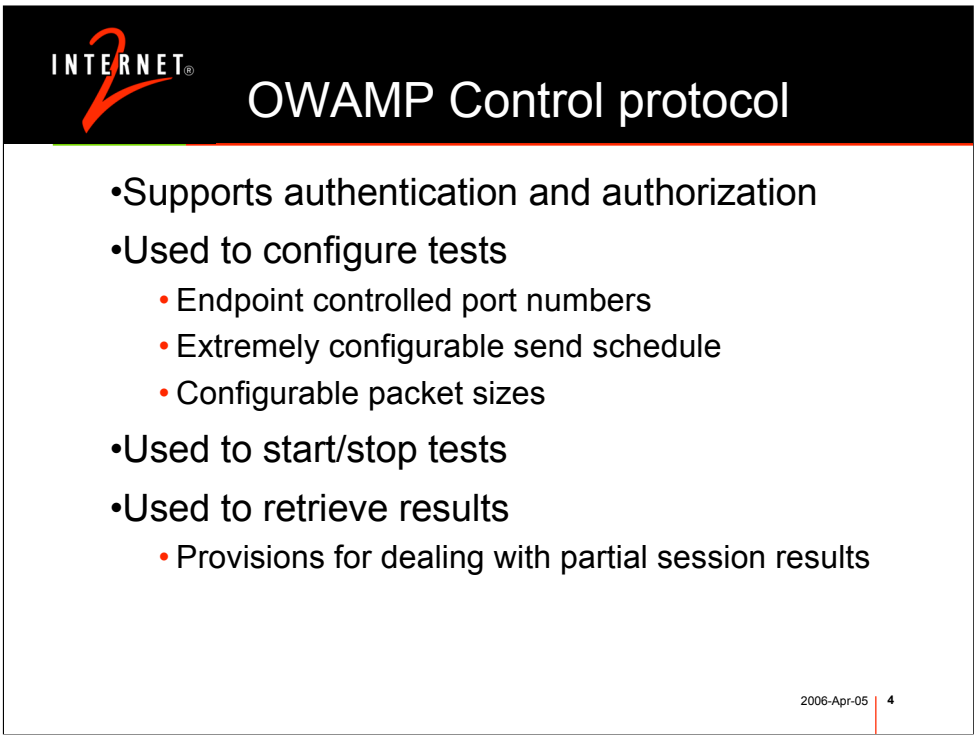
OWAMP itself is a control protocol, a test packet format, and a sample implementation.

Why the OWAMP protocol?

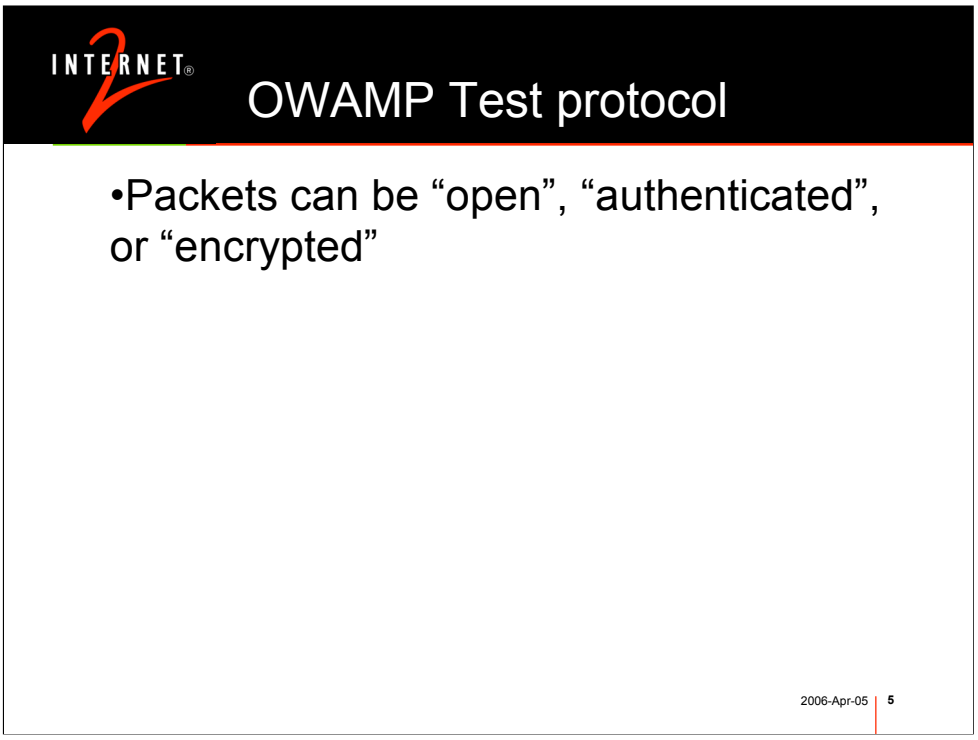
- Find problems in the network
  - Congestion usually happens in one direction first…
  - Routing (asymmetric, or just changes)
  - SNMP polling intervals mask high queue levels that active probes can show
- There have been many implementations to do One-Way delay over the years (Surveyor, Ripe…)
  - The problem has been interoperability.
  - http://www.ietf.org/internet-drafts/draft-ietf-ippm-owdp-014.txt

2006-Apr-05 | 3

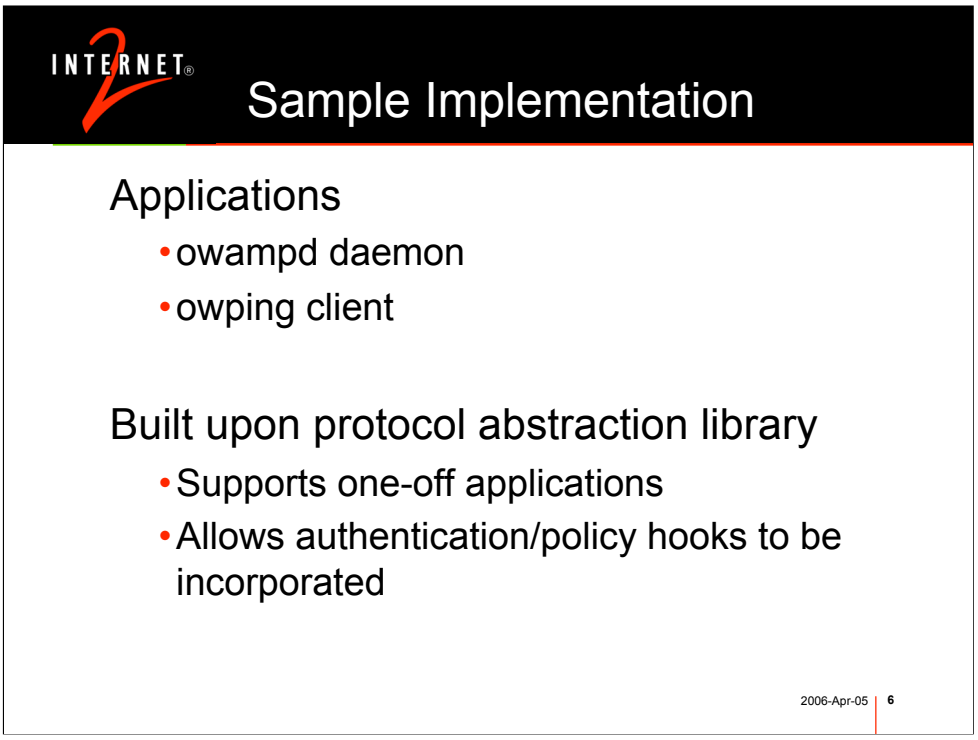
The solution is standards.



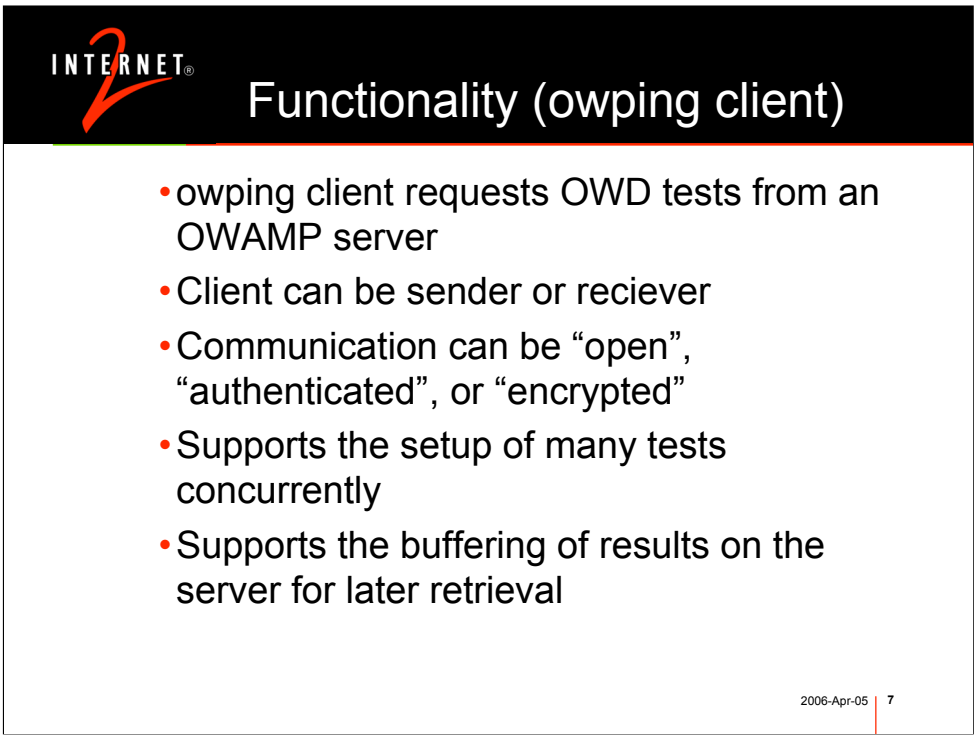
Basically the client makes requests for tests with a server.



The protocol is effectively a packet format.



There is a one-off application in the distribution called powstream that creates a perpetual stream of packets from the server to the client for continuous testing.



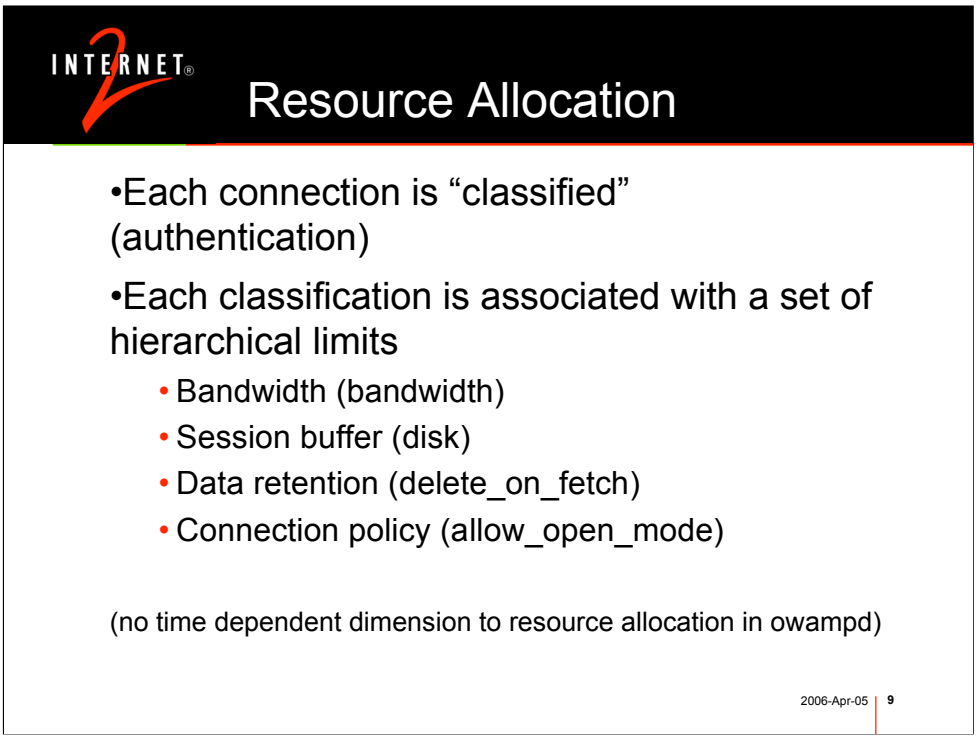
The command-line arguments were made as similar to ping as possible.

Functionality (owampd)

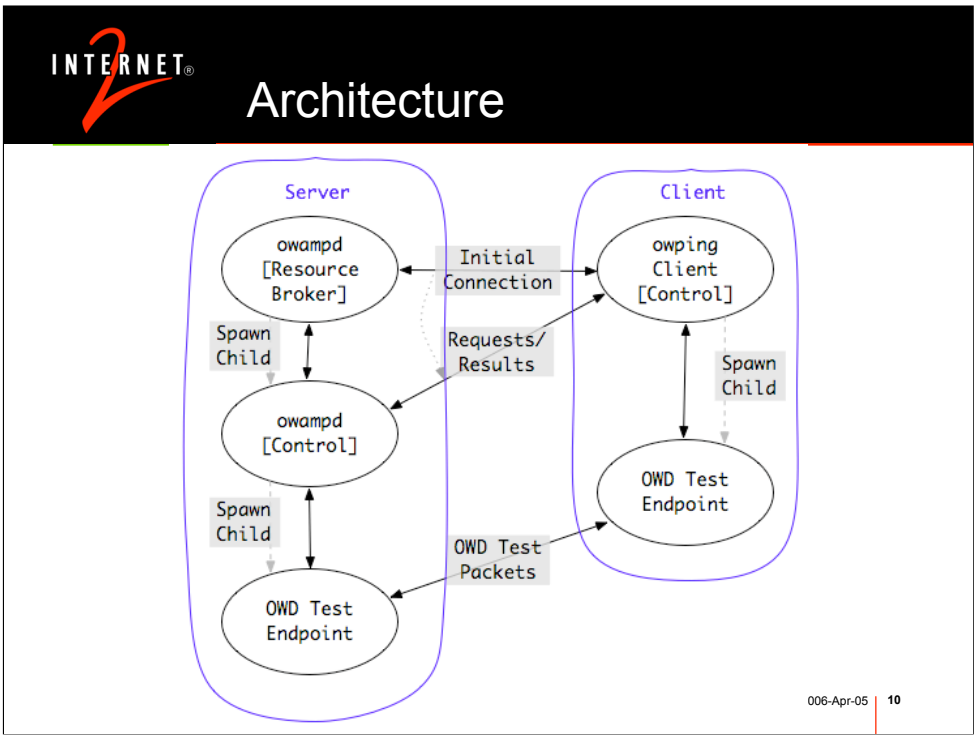
owampd

- Accepts requests for OWD tests
- Responds with accepted/denied
- Tests are formally started with a StartSessions message from the client.
- Runs tests
- Sessions with packets received at the server are buffered for later retrieval

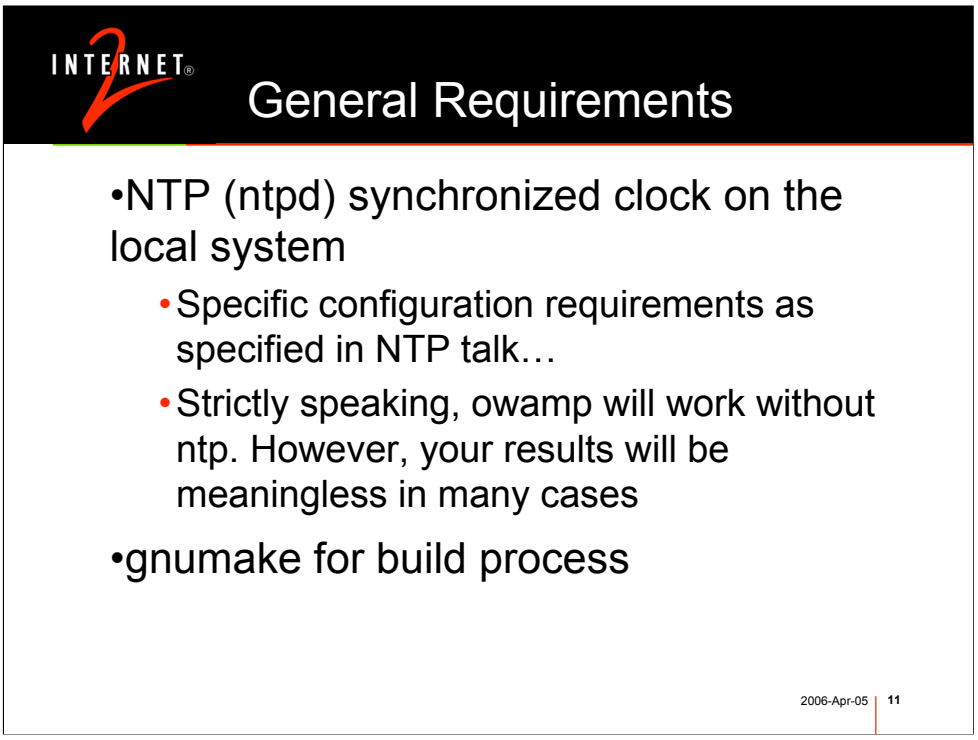
Standard accept/fork style unix daemon



The parent owampd keeps track of current resource utilization needed to implement policy.



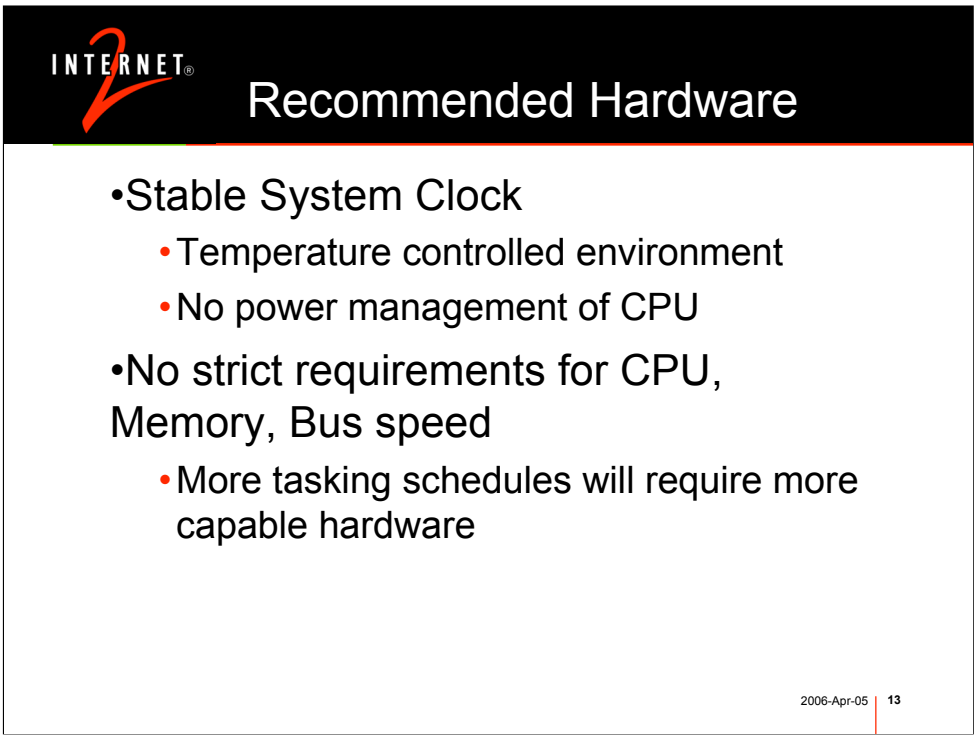
See http://e2epi.internet2.edu/owamp/details.html



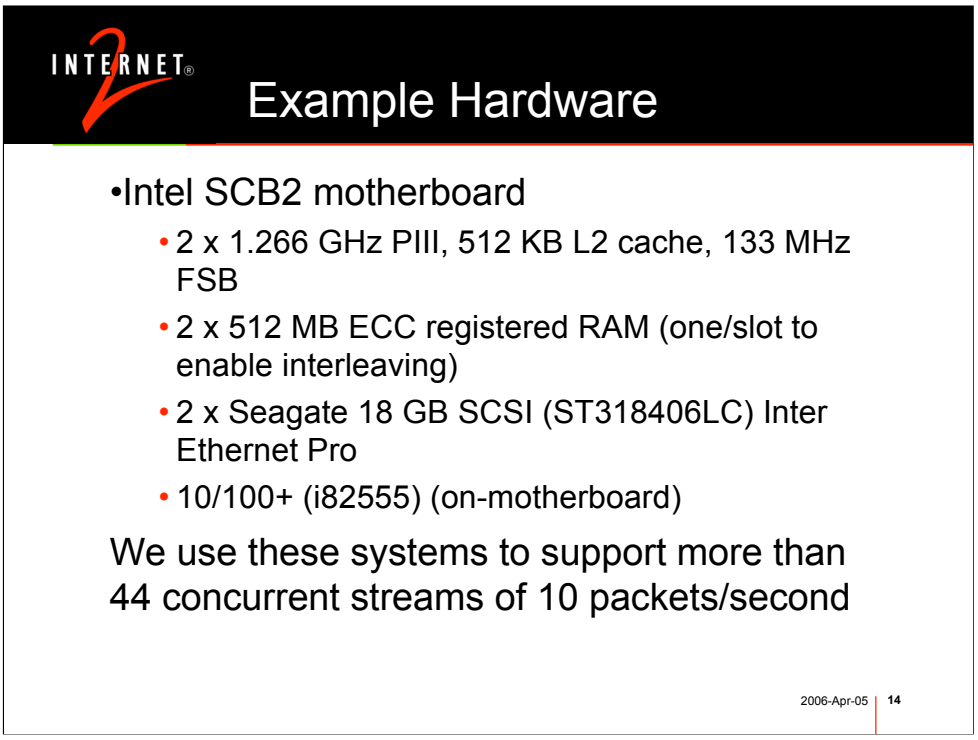
Getting a good stable NTP configuration is the most challenging task for obtaining good owamp results.

Supported Systems

- FreeBSD 4.7+, 5.x, 6.0 (64-bit)
- Linux 2.4, 2.6 (64-bit)
- MacOS X 10.4.5
- Solaris 10.4.5
- (Most recent versions of UNIX should work)



Stable system clock is the most important feature.



The 44 concurrent streams represent intra-abilene testing. Hosts doing tests with external hosts are doing more.

Operational concerns

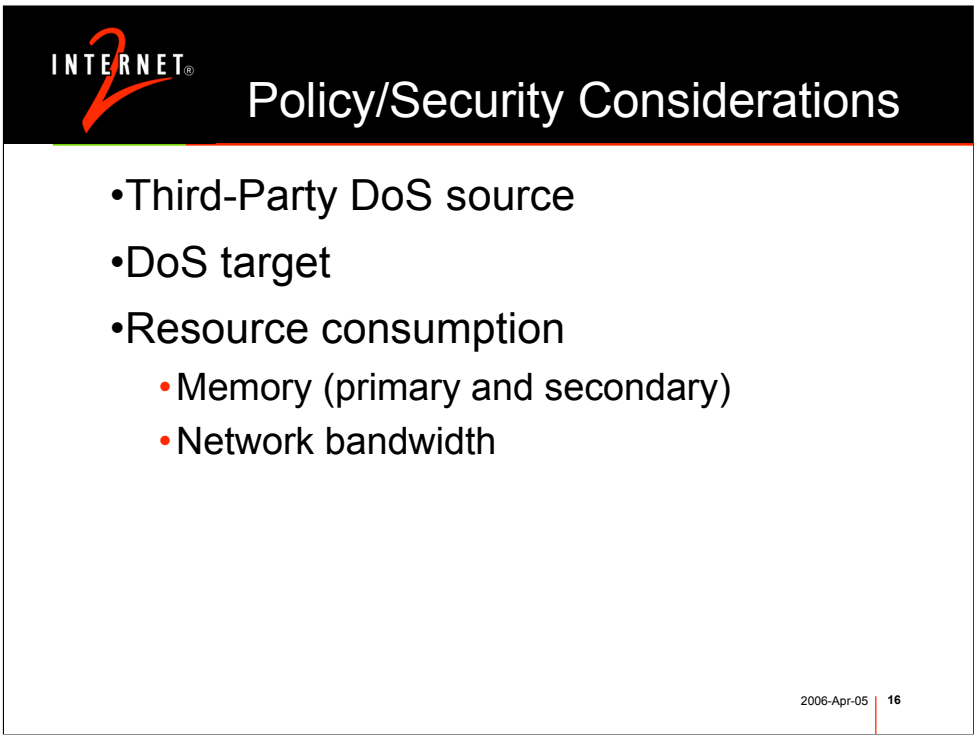
Time:

- NTP issues predominate the problems
- Determining an accurate timestamp “error” is in many ways more difficult than getting a “very good” timestamp
- Working as an “open” server requires UTC time source (For predefined test peers, other options available)

Firewalls:

- Port filter trade-off
  - Administrators like pre-defined port numbers
  - Vendor manufactures would probably like to “prioritize” test traffic
  - Owampd allows a range of ports to be specified for the reciever

These issues will be discussed further in the hands-on session.



DoS source:

A compromised owampd server could be used to send packets toward others. The implementation ensures that sessions can not be directed to random hosts in unauthenticated mode. (Only toward the OWAMP-control client.) Reasonable bandwidth limits and well protected AES keys should limit this risk.

DoS target:

Packets directed toward an owampd server can/will affect the precision of the valid test traffic.

Someone might try to effect data plots by targetting hosts that do owamp measurements.

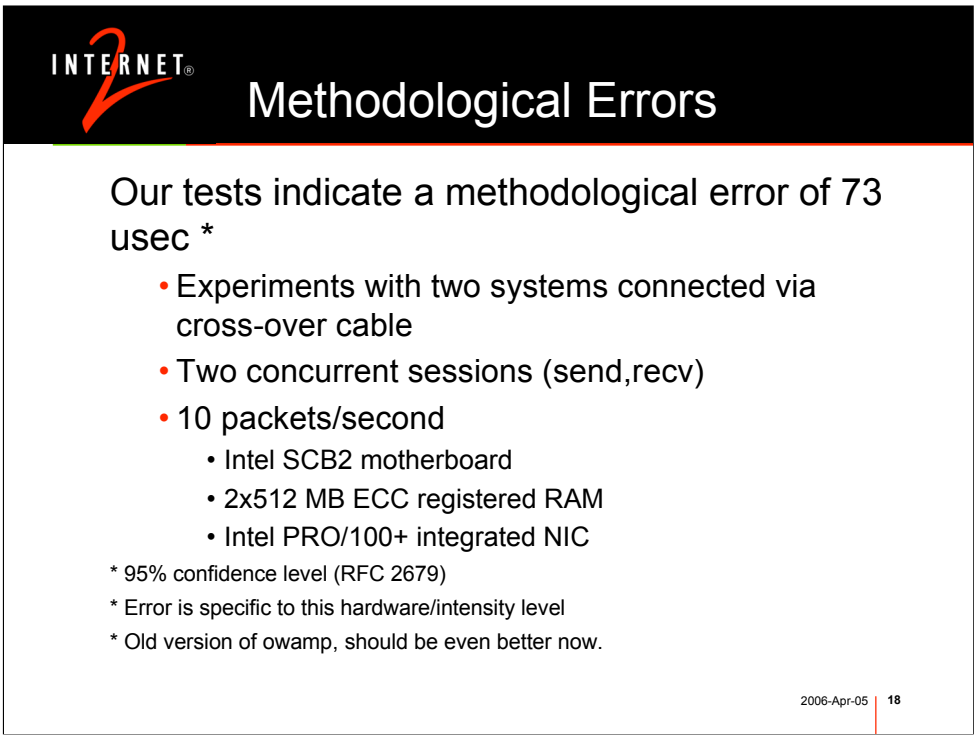
Resource Consumption:

owampd has policy controls to allocate resources to appropriate users.

Policy Recommendations

- Restrict overall bandwidth to something relatively small
  - Most OWAMP sessions do not require much
- Limit “open” tests to ensure they do not interfere with precision of other tests

On Abilene, we attempt to be open until we can’t.



You should expect your results to be valid within 100usec's of the error reported. (The error reported represents the NTP error, but does not include this error.)

Availability

- Currently available

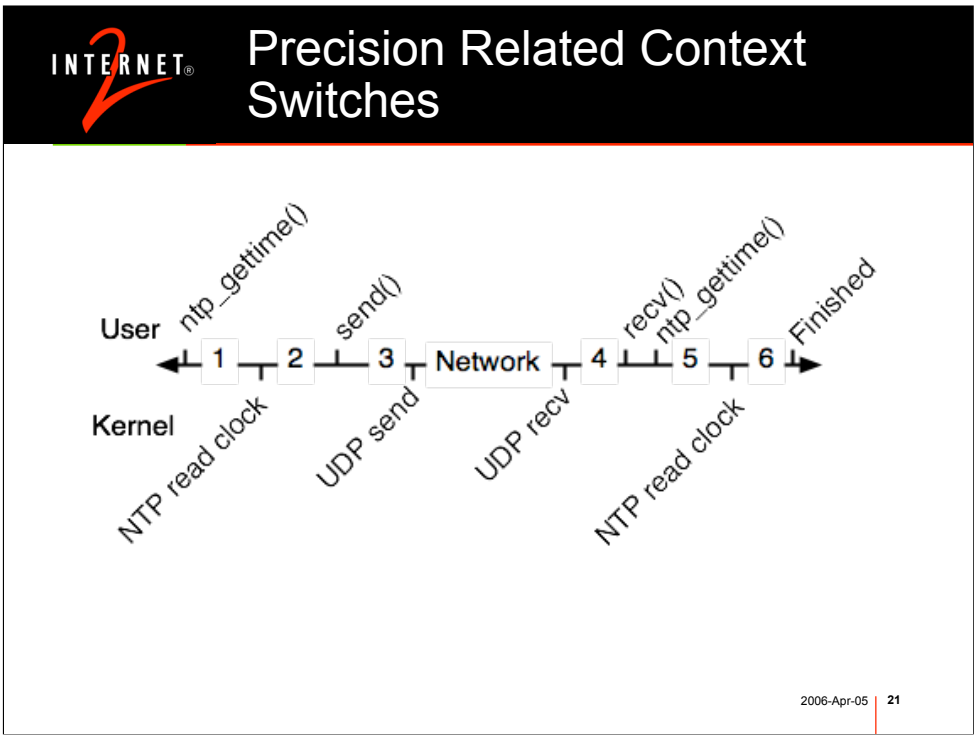
  http://e2epi.internet2.edu/owamp/

Mail lists:

- owamp-users@internet2.edu
- owamp-announce@internet2.edu

https://mail.internet2.edu/wws/lists/engineering


INTERNET®
www.internet2.edu



This slide just illustrates where most of the methodological error comes from, it is not strictly important to understand from an operational point of view.

There are 6 context switches from when the sender process first requests the time until the receiver process has all the data.

The context switches that affect precision are 2–5 because they occur between the fetching of the system clock values on the respective systems.

Implementation decision was made to do timing in user space to allow for more portability. (custom kernel's not required)

> It would be possible to completely remove these 4 context switches if the implementation created its own system call that combined the functionality of reading the timestamp with the sending and receiving of the UDP packet.
>
> This is a possible future enhancement that could be done to improve worst-case accuracy for the subset of installations that need more precision.

Don't go into this unless there is interest and questions.