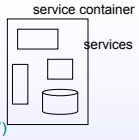
Advanced Services

Roy Williams
California Institute of Technology

Building Compute Services



- Developer and Admin
  - Services should be built by developers
  - In a framework managed by an adminstrator
- Service developers must be careful
  - Services can be dangerous (eg "execute any command")
- Service users authenticated with "graduated security"
  - Easy to start, but great power is possible
  - Or just keep it anonymous
- Asynchrony for compute intensive jobs
  - Jobs submitted to batch queue
  - Unique benchID may be used to monitor job & return results
- From "clicking" to "scripting"
  - Services may be accessed by clicking on a web page or with scripted client codes
  - Authentication for web clicking comes from a certificate in browser
  - Scripted access requires a certificate

Persistent Storage ("workbench")

Ceramics class meets each week for 8 weeks

Workbench

- Persistent storage
- Just a directory in the web space
  - Initiated by service
  - Tools operate on files in workbench
    - http://.....?bench=39840422 & action=PCA & (other params)

```
[roy-williams-computer:WebServer/Documents/benches] roy% ls -l
total 0
drwxr-xr-x   13 www  admin  442 Jun 29 14:50 1621307736892568800492040943?160
drwxr-xr-x    7 www  admin  238 Jul 21 12:36 2349464573581719588772382683462?
drwxr-xr-x    7 www  admin  238 Jul 24 11:04 5670532085712293?8827685654?9833
drwxr-xr-x    7 www  admin  238 Jul 21 12:36 7?66078357?1044435?582937347000
drwxr-xr-x    7 www  admin  238 Jul 21 16:45 74291586152694000084049339888583
[roy-williams-computer:WebServer/Documents/benches] roy% ls -l 1621307736892568800492040943?160
total 1736
-rw-r--r--   1 www  admin   67904 Jun 20 19:08 GSC2.votable
-rw-r--r--   1 www  admin   96388 Jun 20 19:05 cutouts.votable
-rw-r--r--   1 www  admin   66466 Jun 20 19:08 ned.votable
-rw-r--r--   1 www  admin   48200 Jun 20 19:09 sdss.votable
-rw-r--r--   1 www  admin    2953 Jun 20 19:01 sources.csv
-rw-r--r--   1 www  admin  352267 Jun 29 14:50 sources.votable
-rw-r--r--   1 www  wheel  121696 Jun 29 14:46 spectra.votable
-rw-r--r--   1 www  admin  118419 Jun 21 09:42 twomass.votable
[roy-williams-computer:WebServer/Documents/benches] roy%
```

Workbench

- URL to workbench is obscure
  - http://localhost/cgi-bin/vim?benchID=162130773689256880049204094371 60
  - Can send to your colleague
- Set up as
  - Read is free but URL is obscure
  - Using tools / write permission via password
- Reaping
  - Maybe 30-day lifetime for workbench storage?
    - Need cron process to delete old benches

Keywords

- “bench”
  - If present, specifies workbench
- “action”
  - What should the server do?
    - Create workbench (provide password)
    - Upload data
    - Start algorithm
    - Monitor run (does the result exist?)
    - Download result
- Others:
  - Depends on action, specifies detail

VIM server

```
if actionkey == "init":
          benchID = bench.makeBench()

elif form.has_key("bench"):
          benchID = form["bench"].value

else: print "No bench specified -- exiting"

# bench must be 32 decimal digits (NOT ../../precious)
if re.match(r'^[0-9]{32,32}$', benchID, re.IGNORECASE) == None:
          print "Sorry, but %s does not look like a valid benchID name" % benchID
          sys.exit(1)

bench.setBenchID(benchID)

if actionkey == "urltable":    actions.urltable(bench)
if actionkey == "deletetable": actions.deletetable(bench)
if actionkey == "fetch":       actions.fetch(bench)
if actionkey == "addcol":      actions.addcol(bench)
if actionkey == "select":      actions.select(bench)
if actionkey == "join":        actions.join(bench)
if actionkey == "sort":        actions.sort(bench)
```

Making things easier

- Let them log in!
  - Keeps record of workbenches
  - Who owns which
  - Users can ask for “my workbenches”
  - Can make log for funders
    - Who is doing what
- BUT
  - Users *hate* to register at websites

Security and Certificates

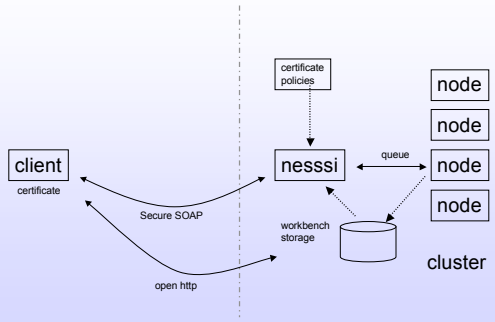
- Stop attacks
- Access to secret data
- Access to big resources
- BUT
  - Lots of extra infrastructure
  - Users hate it

NESSSI

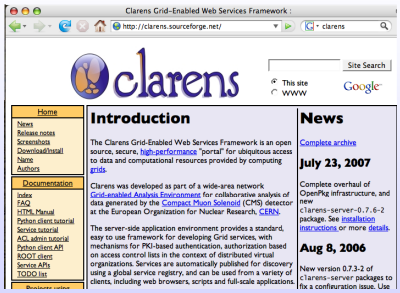
NVO Extensible Secure Scalable Service Infrastructure

- Services are science-oriented
- Services are made by trusted developers from the science community
- Web forms OR command line (Python API)
- Built-in security (X.509 certificates)
- Very large jobs can be run
- Easy to get a certificate
- No complex install needed by client
- Different levels of certificate get different service
- Is installed on Teragrid
- Services can be part of a workflow

Nesssi

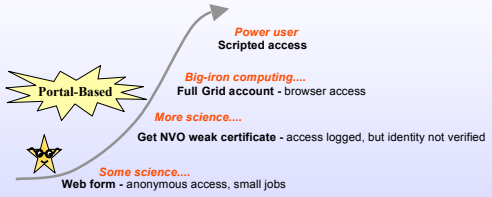


Clarens server

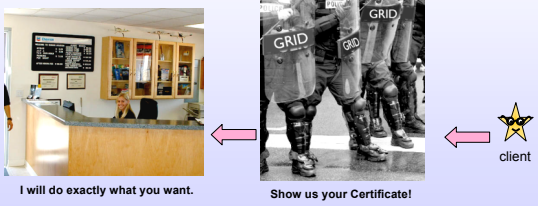


An open-source webserver based on OpenSSL.

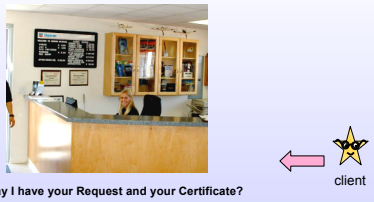
A "Graduated Security" Model



Traditional Grid Security



Graduated Security



Authentication with Certificates

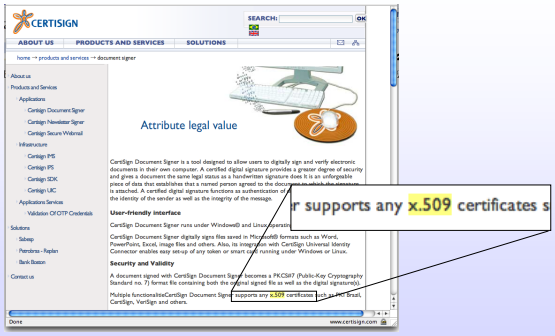
- A digital certificate proves who you are
- X.509
  - Usually encrypted by passphrase
- Certificate as login
  - Map from certificate to account

Certificates

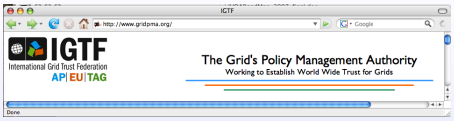
The Virtual Observatory as a Virtual Organization

This is a US driver's licence.
In the US it proves identity strongly.
It is like a **strong certificate.**

This is a loyalty card where I buy food.
(You can put a false address on the application.)
It is like a **weak certificate.**



How to be a Certificate Authority



In order for an RA to validate the identity of a person, the subject should contact the RA **face-to-face and present photo-id** and/or valid official documents showing that the subject is an acceptable end entity as defined in the CP/CPS document of the CA.

In case of host or service certificate requests, the RA should **validate the identity** of the person in charge of the specific entities **using a secure method**. The RA should ensure that the requestor is appropriately authorized by the owner of the FQDN or the responsible administrator of the machine to use the FQDN identifiers asserted in the certificate.

Bench ID

- Identify which job we are talking about
  - 32 character hex string eg
    cb28d0753a7fec9a485981f741d425ec
  - Used to monitor a running job

```
sessionID = nesssiServer.cutout.init()
msg = server.cutout.monitor(sessionID)
```

  - Used to form URL where results appear, eg
    - `http://dtf-test1.sdsc.teragrid.org:8080/clarens/shell/cb/cb28d0753a7fec9a485981f741d425ec/cutouts/index.html`
- If you lose the sessionID, you lose your job

Monitoring a Nesssi job

```
<NesssiMonitor>
<Service>Cutout</Service>
<Uname>ux400560</Uname>
<SessionID>774daf5ef52facc68cb03db4b1fdc815</SessionID>
<Sandbox>http://dtf-test1.sdsc.teragrid.org:8080/
        clarens/shell/77/774daf5ef52facc68cb03db4b1fdc815</Sandbox>
<Result>http://dtf-test1.sdsc.teragrid.org:8080/
        clarens/shell/77/774daf5ef52facc68cb03db4b1fdc815/cutouts/index.html</Result>
<QueueStatus>149.envoy.cacr.calte roy      batch    C8845cb     11516     1  --    --  60:00 R   --
    </QueueStatus>
</NesssiMonitor>
```

service name

running as this user

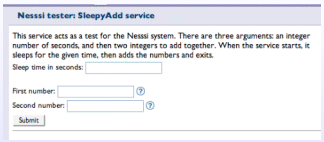
session ID

sandbox URL

results URL

queue status
(R = running)

Example: SleepyAdd



web portal

```
nesssiServer=nesssi.client('https://dtf-test1.sdsc.teragrid.org:8443/clarens/',debug=0)

sessionID = nesssiServer.sleepyadd.init()
print "Your session ID is", sessionID

# Run: sleep 30 seconds then add 52 and 344
nesssiServer.sleepyadd.run(sessionID, "-time 30 -n 52 -m 344")
```

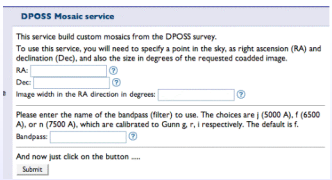
command line

Monitoring the Run

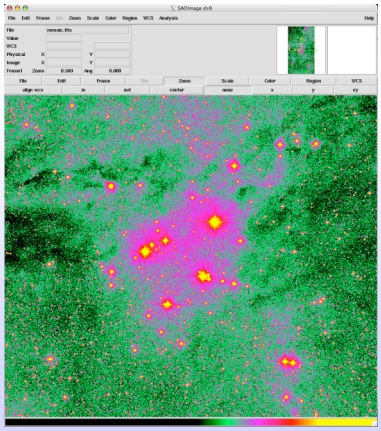
```
<NesssiMonitor>
<Service>Sleepyadd</Service>
<Uname>ux400560</Uname>
<SessionID>a3a167a383111c0cbd6941325b8659aa</SessionID>
<Result>http://dtf-test1.sdsc.teragrid.org:8080/clarens/shell/a3/a3a167a383111c0cbd6941325b8659aa/batch.out</Result>
<Sandbox>http://dtf-test1.sdsc.teragrid.org:8080/clarens/shell/a3/a3a167a383111c0cbd6941325b8659aa</Sandbox>
<QueueStatus>305875.dtf-mgmt1.sds ux400560 dque     Ca3a167       --      1  --    --  18:00 Q   --
</QueueStatus>
</NesssiMonitor>
```

Key n is 52
Key m is 344
Key time is 30
Sleeping for 30 seconds
Waking up...
Sum of 52 and 344 is 396

Mosaic Service

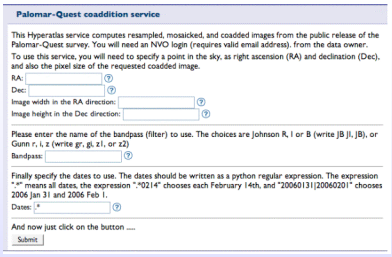



```
nesssiServer=nesssi.client('https://envoy.cacr.caltech.edu:8443/clarens/',debug=0)

mosaic_loc = "-ra 49.1 -dec 60.1 -rawidth 0.5 -decwidth 0.5 -filt f -bgcorr 0"

session = nesssiServer.dpossMosaic.mosaic(mosaic_loc)
print "Your session ID is %s." % session

msg = dbsvr.dpossMosaic.monitor(session)
print msg
```

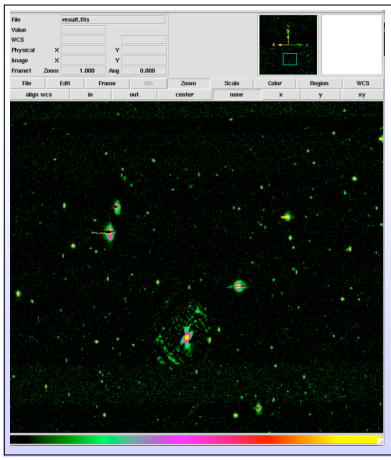


```
nesssiServer.
dpossMosaic.mosaic (
"-ra 49.1
-dec 60.1
-rawidth 0.5
-decwidth 0.5
-filt f
-bgcorr 0")
```

Coadd Service

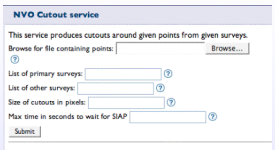



```
nesssiServer=nesssi.client('https://envoy.cacr.caltech.edu:8443/clarens/',debug=0)

# Initialize the service
sessionID = nesssiServer.hyperatlas.init()
print "Session id is ", sessionID

# Arguments for service, the coaddition to do
args = "-bandpass z1 -ra 170.08 -dec 13.275 -rawidth 1.0 -decwidth 1.0"
```



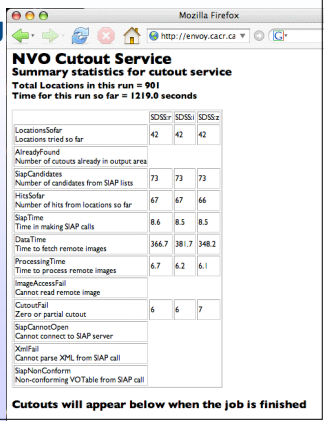
```
-bandpass z1
-ra 170.08 -dec 13.275
-rawidth 1.0 -decwidth 1.0
```

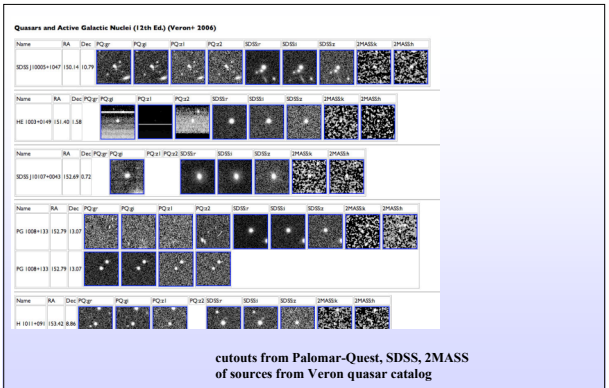
Cutout Service




```
nesssiServer=nesssi.client('https://envoy.cacr.caltech.edu:8443/clarens/',debug=0)
sessionID = nesssiServer.cutout.init()
print "Session id is ", sessionID

# Upload locations file
remoteinputfile = "/shell/%2s/%s/inputfile.xml" % (sessionID[0:2], sessionID)
nesssiServer.upload_file(inputfile, remoteinputfile)

# Arguments for service, surveys to use and cutout size
args  = "-surveys PQ:gr,PQ:gi,PQ:z1,PQ:z2,SDSS:r,SDSS:i,SDSS:z,2MASS:k,2MASS:h "
args += "-size 64"

# Run service
nesssiServer.cutout.run(sessionID, args)
```

Cutout Monitoring





cutouts from Palomar-Quest, SDSS, 2MASS of sources from Veron quasar catalog

Amazon Grid

(who will pay?)

Amazon amazon web services™

- Simple Storage Service

- Write, read, and delete.
- Each object has a unique, developer-assigned key.
- Authentication mechanisms. Objects can be private or public. Rights can be granted to specific users.
- REST and SOAP interfaces
- Default download protocol is HTTP. BitTorrent(TM) also available.

Amazon amazon web services™

- Elastic Compute Cloud

- Create an Amazon Machine Image (AMI) containing your applications, libraries, data and associated configuration settings.
- Upload the AMI into Amazon Simple Storage Service.
- Configure security and network access.
- Start, terminate, and monitor as many instances of your AMI as needed.
- Pay for the instance hours and bandwidth that you actually consume.

- $0.10 per instance-hour consumed
- $0.20 per GB of data transferred outside of Amazon
- $0.15 per GB-Month of Amazon S3 storage

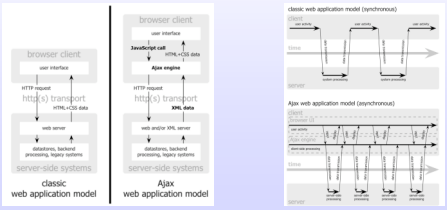
Amazon amazon web services™

- Simple Queue Service

- Move data between distributed application components performing different tasks, without losing messages or requiring each component to be always available.
- Unlimited number of queues, unlimited number of messages.
- New messages can be added at any time.
- A computer can check a queue at any time for messages waiting to be read.
- REST, SOAP and query interfaces.
- The queue creator determines which other users can write to or read from the queue.

AJAX (Asynchronous Javascript + XML)

- Uses browser's XML support: DOM, XSLT
- XMLHttpRequest
- Google Maps is best-known AJAX application



What do GET/POST services lack?

- Format method for describing interface contract
- Reliable messaging
- Digital signatures
- Message routing
- Resource life cycle management
- Asynchronous event notification
- Other capabilities captured by WS-* specs

What is SOAP?

- Simple Object/Service-Oriented Access Protocol (Snakes On A Plane?)
- An XML-based communication protocol and encoding format for exchanging structured information in a decentralized, distributed environment
- W3C specification (http://www.w3.org/TR/soap)

Anatomy of a SOAP message

- An envelope to encapsulate data which defines formatting conventions for describing the message contents and routing directions: header and body
- A message exchange pattern: request/response (RPC mechanism), fire-and-forget
- A transport or binding protocol
- Data encoding rules for describing the mapping of application-defined datatypes into an XML tag-based representation

SOAP example

Request:

```
<soap:Envelope xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
      xmlns:xsd=http://www.w3.org/2001/XMLSchema
      xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
      <soap:Body>
            <ComovingLineOfSight xmlns="http://skyservice.pha.jhu.edu">
                  <z>float</z>
                  <hubble>float</hubble>
                  <omega>float</omega>
                  <lambda>float</lambda>
            </ComovingLineOfSight>
      </soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
      <soap:Body>
            <ComovingLineOfSightResponse xmlns="http://skyservice.pha.jhu.edu">
                  <ComovingLineOfSightResult>float</ComovingLineOfSightResult>
            </ComovingLineOfSightResponse>
      </soap:Body>
</soap:Envelope>
```

Client Invocation Models

- Static: use generated stubs:
  java org.apache.axis.wsdl.WSDL2Java <wsdl url>
- Dynamic:
  - no generated code
  - a proxy dynamically generates a class at runtime that conforms to a particular interface, proxying all invocations to a single 'generic' method
  - Examples:
    - Java : use javax.xml.rpc.Service.getPort() and createCall()
    - .NET : use RealProxy class (must extend ContextBound) or Reflection.Emit
- Generic SOAP client: http://soapclient.com/soaptest.html

Why is SOAP better?

- Asynchrony
- Reliable messaging (e.g. once-and-only delivery, guaranteed or exact execution)
- Send and receive complex datatypes to invoke a particular method not just key-value pairs
- Security
- Binds to other protocols
- Service description

Take a REST from SOAP?

- IVOA jumped into SOAP services in 2002
- But SOAP is perceived as "difficult"
  - WSDL (formal service description) is complex and not interoperable
- REST and GET are perceived as easier
- Where is the sophistication of SOAP really needed?

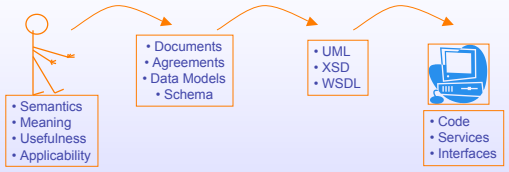
Steering the VO ship

- Short term Pragmatism
  - useful tools now
  - simple protocols (eg cone search)
  - "just use RA and Dec"

vs

- Long term Architecture
  - modular suite of interoperable tools
  - sophisticated protocols (eg skynode)
  - sophisticated Space-Time coordinates

Building Information Standards



A Data Model is a bridge from community to computers

Questions?