



GridPP
UK Computing for Particle Physics

AutoSD - Automatic Service Discovery

Abdeslem DJAOUI

June 29, 2006

Queen Mary University of London



Who needs SD?

- Batch job submission - GGF HPC profile
 - Manual configuration, No need for SD
- Semi-static deployments
 - Tightly Controlled environment
 - Mostly manual configuration, some dynamic SD
- Self-configuring services and self-healing applications
 - Highly dynamic environment, Mostly Automated configuration and fail-over with some manual configuration
 - SD as critical as network



1. How to find information service in the first place
 - Bootstrapping
 2. How to register service ID and description
 - What information/data model
 3. Type of queries
 - Polling (inefficient) versus subscriptions
 - Query expressiveness, correctness, semantics, ..
 4. Robustness, availability, scalability, ...
 - Initial design does not always allow them
- 1 is still a hole, 2 and 3 well advanced, 4 started to be addressed but not to the satisfaction of users



- EGEE User requirements on bootstrapping
 - #100537: There must be a simple deterministic procedure to find the entry points to the information system (bootstrapping problem).
 - #100538: It must be possible to obtain information about all grid services the user is authorized to use (in particular, VOs, RBs, CEs, SEs...) through a well known and unique procedure. The information system bootstrapping issue (see requirement #100537) should be solved for this. For the sake of simplicity, there must be a well known and accessible entry point to the information system.
- Discussed on many occasions but no solution identified
- Current SD is just an abstracted API on top of existing Info systems
 - Does not do much for facet 1 and 4



- Subscription using the network infrastructure
 - Senders transmit one UDP datagram packets to a group address
 - Receivers join the multicast group
 - Network routers and switches send a copy of packet to each members of the group
- Spontaneous zero-configuration discovery of services
 - Services/Resources send an announcement to a multicast group when they are started and when they are stopped
 - Client listen to the multicast group and discover services as they come and go

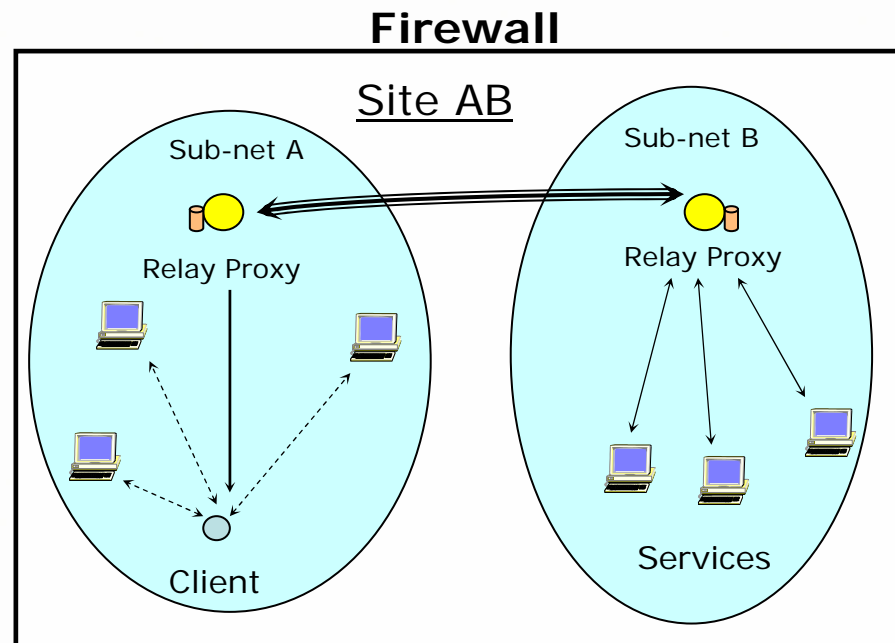


- Client can also probe for listening services based on name, type ... by sending a probe to the group address
- Multicasting relies on existence of IP network only
 - No need to know an information service endpoint before discovering services of interest
- Seemingly the ideal solution to bootstrapping problem
 - But standard router configurations limit effectiveness of Multicasting
 - Usually limited to sub-net



Beyond a sub-net

- Overlay network of Relaying Proxy services
- Configured manually or using DHCP, ...





- Resource Description Framework a W3C recommendation since 2004
 - Increased activity and availability of tools
- Natural model for SD
 - Intended for describing and discovering resources on the web
- RDF vocabularies can be augmented with ontologies to add semantics to descriptions
 - If B *isReplicaOf* A, C *isReplicaOf* B and *isReplicaOf* property is declared to be *symmetric* and *transitive* -> A is replica of C



AutoSD and gLite

- A complement to R-GMA, BDII and SD API
- Current SD API in gLite can use AutoSD for *bootstrapping*
- AutoSD enables automatic configuration and failure recovery
- AutoSD makes a distinction between Local sub-net, Local site information and information from wide area network
 - Different security requirements - eased locally
- Use of UDP has some advantages
 - Many services/applications can listen on the same port



- User connects to a network using a laptop and can find where local UI, RGMA servers ... are?
- UI can keep the list of its services up to date automatically
- Since only one R-GMA site publisher, it may wish to use AutoSD to advertise its existence
- Load Balancing: Adding new resources can be discovered automatically
- Always-on Service Discovery even
 - When network link to outside is broken
 - When Grid Information system unavailable



- Existing prototype in java
 - Proof of concept
 - Unreliability of UDP not manifested during tests on sub-net (with small packet sizes < 512 bytes)
 - outstanding bug in JVM when mixing IPv4 and IPv6 hindered progress
 - Work around found
- Working with UDP has advantages and disadvantages
 - More than one client/service can listen on a given port
 - AutoSD has to deal with all packets received on given port
 - Complicated as number of protocol messages increases



- Improve AutoSD design to ease integration with existing Info systems
 - Minimize coding for services and applications that use AutoSD
- Align AutoSD RDF model with GLUE schema
- Change current ad-hoc protocol to new RDF
- Deploy, test and produce first release
- Work on relaying proxies



- We are finally addressing long standing requirements on SD which we couldn't solve with current system
- AutoSD will enable move away from manual configuration and manual failure recovery
- AutoSD complements existing Info systems and add resilience to the system
- AutoSD is designed to allow more to be easily added in the future - e.g Semantics