



INTERNET SOFTWARE CONSORTIUM

# Registry Workshop

AFNOG 2003, Kampala, Uganda

Joe Abley <jabley@isc.org>

# Theory

- What is a DNS Registry?
- Interaction with Others
- Registry Policy
- The Extensible Provisioning Protocol (EPP)
  - just a taste, not an exhaustive study
- ISC OpenReg
  - architectural overview

# Practice

- ISC OpenReg
- Unpack, install and configure an ICANN-model registry
- Publish a zone on a master nameserver
- Interact with the BILL registry using EPP

**What is a Registry?**

- The point of a registry is to publish a zone which delegates child zones to other nameservers
- If you're not in the business of delegating zones to others, you quite possibly don't care about how registries are run
- Registry systems provide a systematic and automated method of maintaining a zone with a limited and well-defined structure

# Registries

- Receive and validate external data
- Store data
- Publish data (DNS, whois, web sites, etc)

# Data In

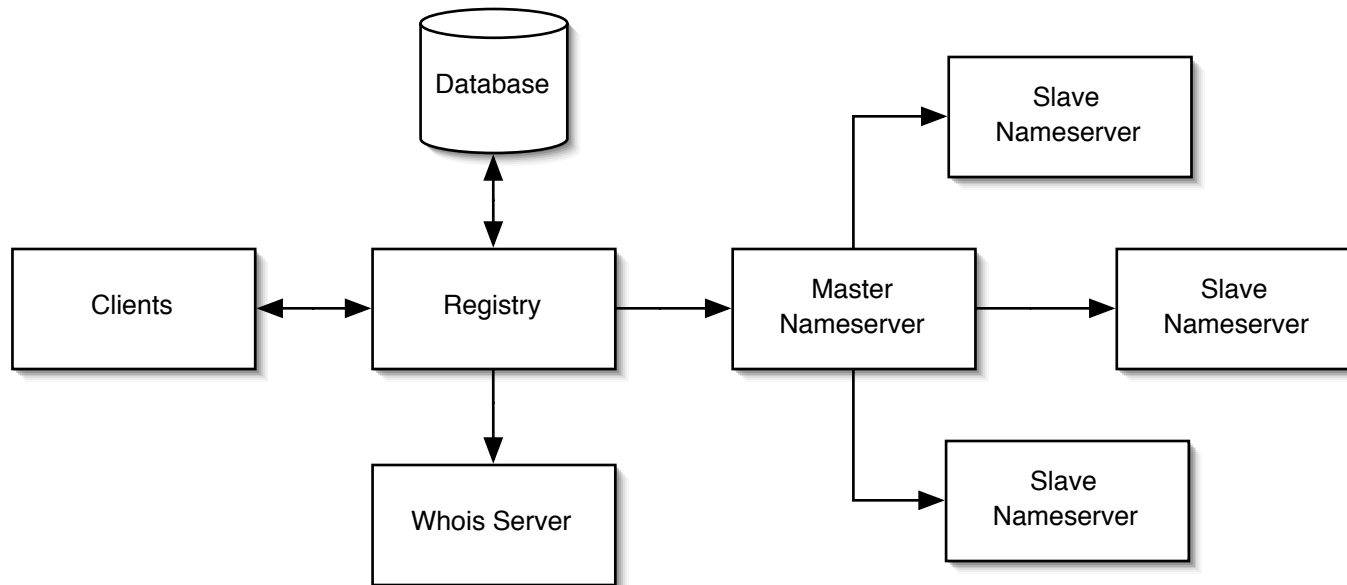
- Domain names
- Nameservers (names, addresses)
- Meta-data
  - authentication
  - technical coordination
  - billing (renewals, payments)

# Data Out

- A Zone File
  - a list of delegations (NS records)
  - delegation glue (A, AAAA records)
  - published via a master nameserver, replicated to slaves
- Whois
- Other Data (statistics, logs)



# Data Flow



# Transactions

- Add and delete records
- Modify records
  - add, delete nameservers (change delegation)
  - change metadata
  - set status attributes

# Manual Registries

- Some registries don't have to process many transactions
  - GOVT.NZ
  - AQ
  - INT
  - NAME (bad joke)
- A registry might just consist of a zone file edited by hand

# The BILL Zone

- We have invented a fictional top-level domain called BILL for the purposes of this workshop
- Right now, the BILL zone is maintained manually
  - send mail to Bert
- Let's start by thinking about what the BILL zone might look like

# The BILL Zone

\$TTL 86400  
\$ORIGIN BILL.

```
@          SOA          A.BILL-SERVERS.BILL. BERT.SECRET-WG.ORG. (
                                2003021801 ; serial
                                1800      ; refresh (30 mins)
                                900       ; retry (15 mins)
                                604800   ; expire (7 days)
                                86400 )   ; negative cache TTL (1 day)

                                NS        A.BILL-SERVERS
                                NS        B.BILL-SERVERS

A.BILL-SERVERS      A        192.168.53.1
B.BILL-SERVERS      A        192.168.59.1

; Duck Corp.
; contact Archibald Featherington, +64 21 625-182, feathers@duck.bill
DUCK                NS        NS1.CLEAR.NET.NZ.
                    NS        YELLOW.DUCK
YELLOW.DUCK         A        192.168.51.1

; Electricity Company Inc
; contact Penrose Moneygrabber, +1 650 779-7000, money@electricity.bill
ELECTRICITY        NS        DNS87.REGISTRAR.COM.
                    NS        DNS136.REGISTRAR.COM.
```

# Transactions

- Add a domain
  - add NS records
  - add glue records (A,AAAA) if necessary
  - store meta-data

```
; Where-Is-Bill Enterprises Ltd
; contact Bill Manning, +1 310 555-8010, bmannings@ep.net
WHERE-IS                NS          FLAG.EP.NET.
                        NS          DOT.EP.NET.
```

# Transactions

- Modify nameservers, metadata
  - change NS records
  - change glue records (A,AAAA) if necessary
  - change meta-data

```
; Where-Is-Bill Enterprises Ltd
; contact Bill Manning, +1 310 555-8010, bmannings@ep.net
WHERE-IS                NS        BANNER.EP.NET.
                        NS        SLASH.EP.NET.
```

# Transactions

- Remove a domain
  - remove NS records
  - remove glue records (A,AAAA) if necessary
  - remove meta-data

```
; Where-Is-Bill Enterprises Ltd
; contact Bill Manning, +1 310 555-8010, bmannings@ep.net
; deleted 2003-02-20 by jabley@isc.org
; WHERE-IS                NS          BANNER.EP.NET.
;                          NS          SLASH.EP.NET.
```



# Registries are Simple

- Relatively small number of transaction types
- Simple zone
- However, even a simple process can become complicated if it has to deal with several hundred transactions per second and provide high availability
- automation can help

# Registry Structure

# Tedious Definitions

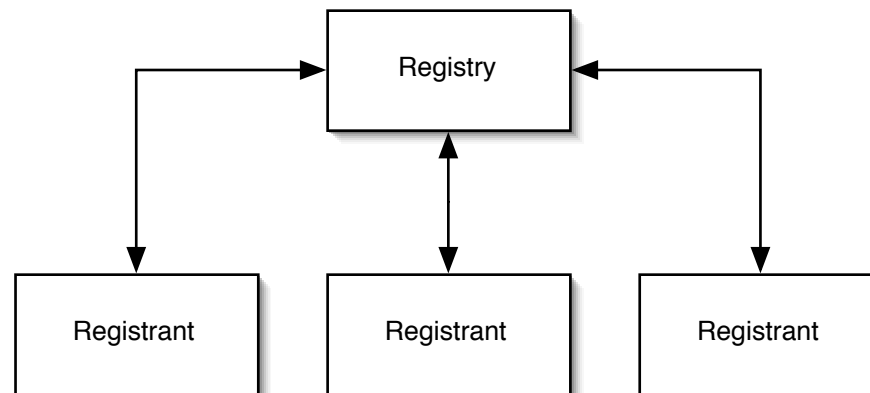
- “registrant” – the organisation or person responsible for a domain
- “registrar” – the middlemen who interact with registries on behalf of registrants
- “registry” – the organisation which maintains the register and publishes the zone
- “register” – the data that is maintained by the registry

# Blame ICANN

- These definitions came about as part of the process by which Network Solutions were divested of some of the responsibilities for running COM, NET and ORG
- The words are very similar
- The words are hence very confusing
  - how convenient

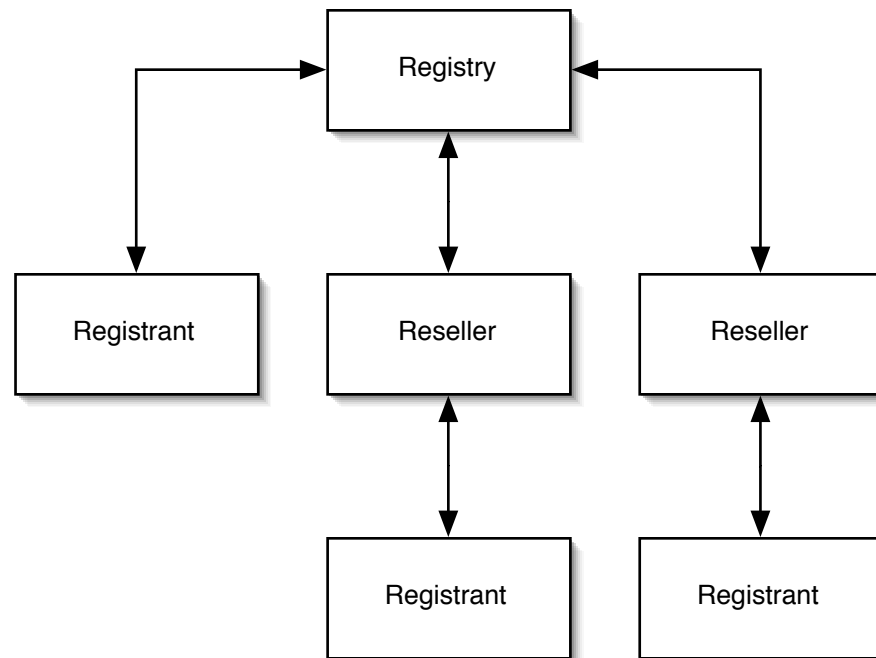
# Simple Registry Model

- Registry deals directly with Registrants
- No Registrars
- “Single Access Registry”
- “Monopoly Registry”

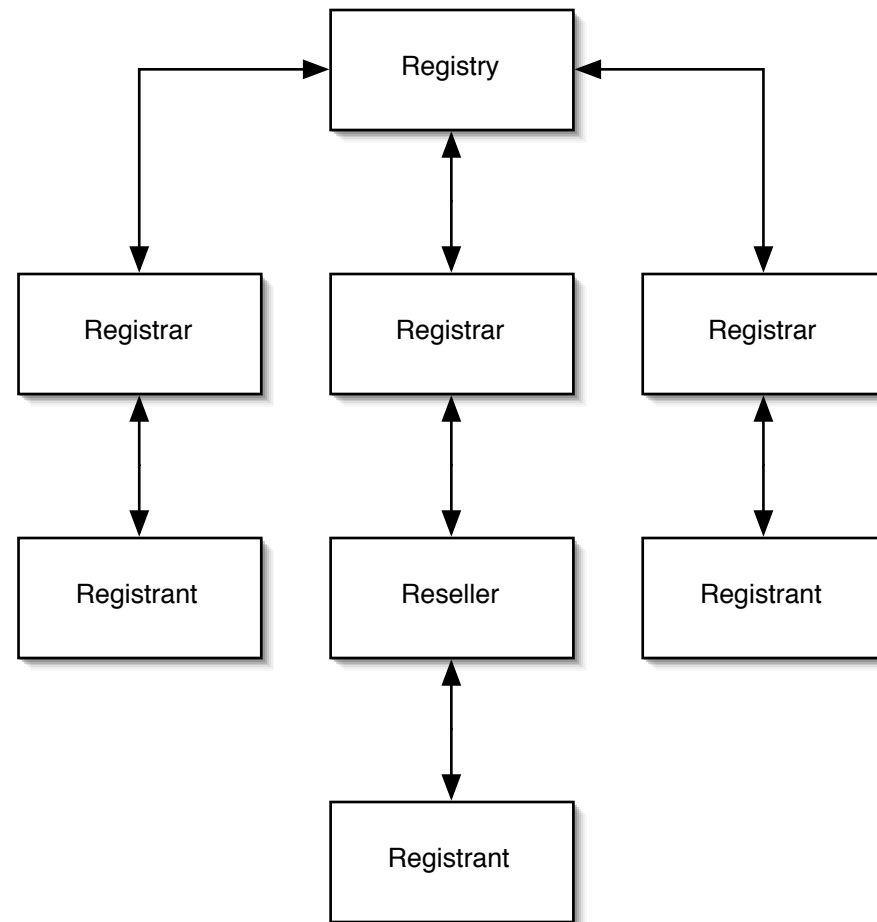


# Simple Registry with Resellers

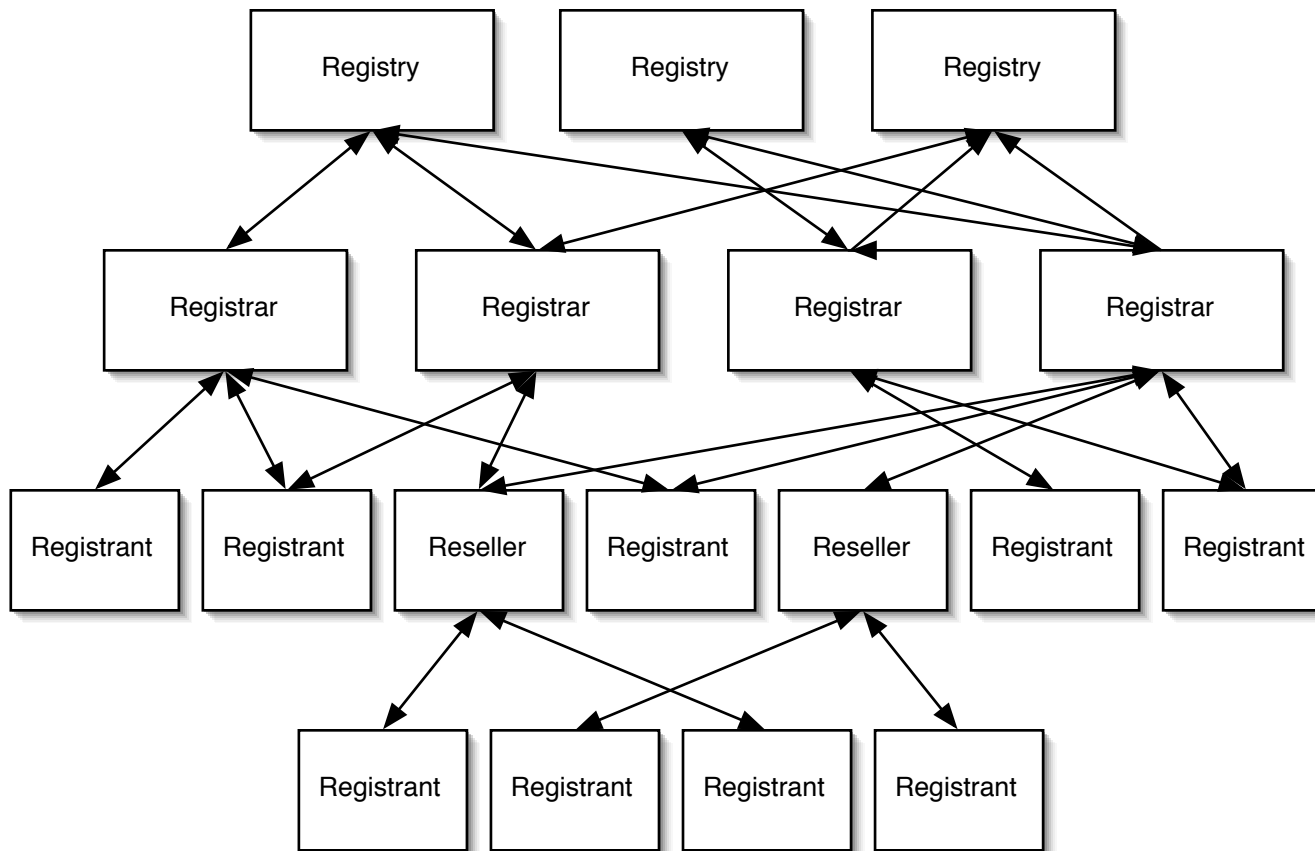
- Registrants can deal directly with the Registry, or they can deal with Resellers



# The ICANN Shared Registry Model



# Glorious in its Simplicity





# Additional Transactions

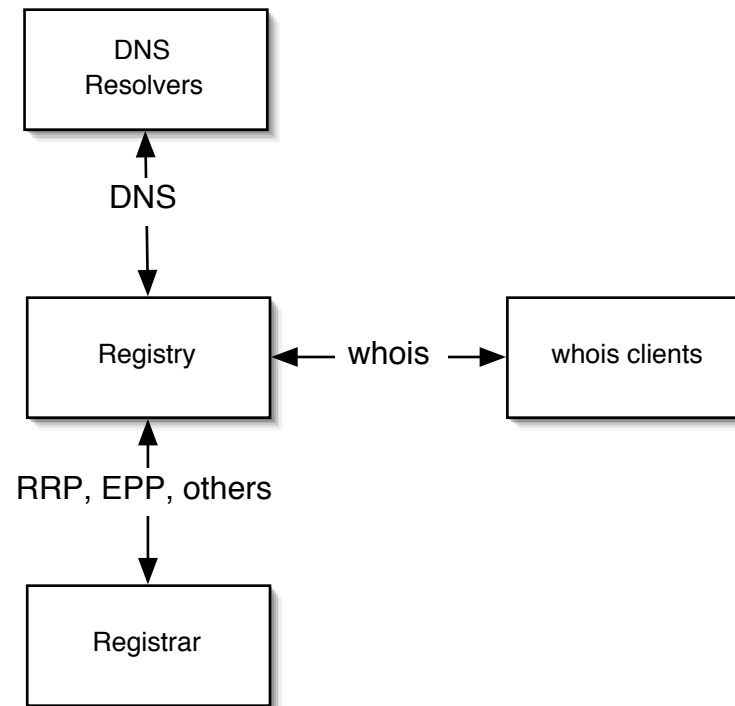
- Registries only interact with Registrars
- For any particular domain, a Registrant only interacts with one Registrar
- That Registrar is said to “sponsor” the domain
  - registry “transfer” transaction

# Thick vs. Thin

- Some shared-registry systems distribute much of the registry metadata to registrars, rather than maintaining it centrally
  - “thin” registry (COM, NET)
- Other shared-registry systems keep all the metadata central
  - “thick” registry (INFO, NZ, CA, etc)

# Interaction with Others

# Contact with Others



# whois

- Mechanism for retrieving metadata from registry
- RFC 954 ("nickname")
  - no data format specified
  - much of the specification describes a query/retrieval strategy that has never been implemented
  - transport protocol is poorly specified
- Every registry's whois output looks different

# whois.crsnic.net

Whois Server Version 1.3

Domain names in the .com and .net domains can now be registered with many different competing registrars. Go to <http://www.internic.net> for detailed information.

Domain Name: VIX.COM  
Registrar: NETWORK SOLUTIONS, INC.  
Whois Server: whois.networksolutions.com  
Referral URL: <http://www.networksolutions.com>  
Name Server: NS1.GNAC.COM  
Name Server: NS-EXT.VIX.COM  
Status: ACTIVE  
Updated Date: 05-nov-2001  
Creation Date: 20-jun-1995  
Expiration Date: 19-jun-2003

>>> Last update of whois database: Wed, 19 Feb 2003 05:39:55 EST <<<

The Registry database contains ONLY .COM, .NET, .EDU domains and Registrars.

# whois.srs.net.nz

```
% New Zealand Domain Name Registry Limited
% Users confirm on submission their agreement to all published Terms
%
version: 0.95
query_datetime: 2003-02-20T02:33:39+13:00
domain_name: patho.gen.nz
query_status: 200 Active
domain_dateregistered: 1997-06-30T00:00:00+12:00
domain_datebilleduntil: 2003-06-30T00:00:00+12:00
domain_datelastmodified: 2003-02-03T17:03:30+13:00
domain_delegaterequested: yes
%
registrar_name: Domainz
registrar_address1: Private Bag 1810
registrar_city: Wellington
registrar_country: NZ (NEW ZEALAND)
registrar_phone: +64 4 918-1740
registrar_fax: +64 4 918-1719
registrar_email: 4service@domainz.net.nz
%
registrant_contact_name: Joe Abley
registrant_contact_address1: 10805 Old River Road
registrant_contact_address2: Rural Route 3
registrant_contact_city: Ontario
registrant_contact_province: Komoka
registrant_contact_postalcode: N0L 1R0
registrant_contact_country: CA (CANADA)
registrant_contact_phone: +1 519 641 3738
registrant_contact_email: jabley@automagic.org
%

[etc, etc, etc]
```

# Thick vs. Thin

- Thick registries return all the registry metadata from one place, in one query (e.g. that NZ example)
- Thin registries return a minimal set of data, usually with a referral to a registrar's whois server (e.g. that COM example)
- if you are lucky, the registrar's whois server might actually work today



# More Whois

- RIRs are registries too
  - IP addresses, ASNs, domains
    - Route Policy (RIPE-181, RPSL)
      - Merit's IRR
- [whois.apnic.net](http://whois.apnic.net), [whois.ripe.net](http://whois.ripe.net),  
[whois.arin.net](http://whois.arin.net), [whois.lacnic.net](http://whois.lacnic.net)
- [whois.ra.net](http://whois.ra.net)

# Registry-Registrar Interaction

- Registry-Registrar interaction does not need to be user-friendly
  - user-friendliness is nominally the registrar's business
- Technical, business-to-business protocols designed for bulk transactions

# Registry-Registrar Protocol (RRP)

- RFC 2832
- draft-hollenbeck-rfc2832bis-03.txt
- Used between Verisign Registry and accredited Registrars
  - thin registry
  - somewhat single-purpose, inextensible
- Interactive, session-oriented stream protocol carried out over SSL

# Extensible Provisioning Protocol (EPP)

- IETF standards-track protocol
- Active working group (provreg)
  - (very peculiar and erratic wg chair)
- Based on the exchange of XML documents
- Can use a variety of transport protocols (BEEP, TCP over SSL, potentially others)
- Extensible

# DNS

- Let's not get so wrapped up in the fun and excitement of registry operations that we forget why we're here:
  - registries publish zones to master servers
  - master servers' data is replicated to slave servers
  - resolvers are referred to authoritative servers, and from there can be referred to registrants' nameservers

# Registry Policy

# Reserved Names

- Some registries place restrictions on what names can be registered
  - “offensive names”
  - names which might cause confusion

# Grace Periods

- Registrant mis-types a domain name when registering it, and wants the error to be corrected
- Payment of an invoice is delayed for a couple of days, and a domain is deleted
- Some registries operate grace periods to accommodate these kinds of issues
- sometimes it's easier and cheaper to be flexible than to punish people for making mistakes



# Renewals

- Some registries automatically renew names unless they have been explicitly cancelled
- Multi-year renewals

# Manual Authorisation

- Lost password
- Administrator e-mail address change
- Administrator no longer works at the company
- Mergers, acquisitions

# Registrar Accreditation

- Technical readiness
- certification testing
- Financial, other criteria
- seek to avoid spontaneous combustion of registrars
- ensure that registrar's customer relationship covers all required legal aspects

# The Extensible Provisioning Protocol (EPP)

# Base Protocol

- draft-ietf-provreg-epp-08.txt
- Provides a mechanism for manipulating generic objects
  - does not specify what those objects are
  - does not specify the transport protocol
- Provides some generic session management primitives

# Base Transactions

- “check” – determine whether an object exists
- “info” – retrieve information about an object
- “create”, “delete”, “update”
- “renew”
- “transfer”

# Session-Control Primitives

- “login”, “logout”
- “hello”, “greeting”

# Buzzword-Compliance

- XML
- Stateful protocol with atomic commands
- Extensible via XML namespaces



# Object Mappings

- Domain Object Mapping
- Host Object Mapping
- Contact Object Mapping

# Domain Mapping

- draft-ietf-provreg-epp-domain-06.txt
- Provides an extension to the base object for dealing with domain objects

# Domain Objects

- References to various contact objects (registrant, admin, tech, billing contacts)
- References to various host objects (nameservers, for delegation)
- Various dates (renewal, last modified, created)
- Various status flags
- Authorisation Information

# Host Mapping

- draft-ietf-provreg-epp-host-06.txt
- Provides an extension to the base object for dealing with host objects

# Host Objects

- Hostnames
- Addresses (IPv4, IPv6)
- Various status flags
- Authorisation Information

# About Hosts and Domains

- The host object/domain object abstraction is based on the Verisign registry
  - other registries don't always handle things that way
- Host and domain records have a carefully-specified relationship

# Contact Mapping

- draft-ietf-provreg-epp-contact-06.txt
- Provides an extension to the base object for dealing with contact objects

# Contact Objects

- Names
- Postal Addresses
- E-mail addresses
- Phone, Fax numbers
- Authorisation Information



# Transport over TCP

- `draft-ietf-provreg-epp-tcp-06.txt`
- Defines a well-known port, specifies encryption and wire encoding of XML request and response documents

# IETF Clues

- Provisioning Registry Protocol (provreg) working group
- All the internet-drafts mentioned are current, and are nearing publication as standards-track RFCs
- <http://www.ietf.org/html.charters/provreg-charter.html>
  - mailing list info, archives
  - drafts, milestones, etc
  - open to anybody who is interested

# EPP Examples

# Client Request

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
epp-1.0.xsd">
  <hello/>
</epp>
```

# Server Response

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
  epp-1.0.xsd">
  <greeting>
    <svID>Example EPP server epp.example.com</svID>
    <svDate>2000-06-08T22:00:00.0Z</svDate>
    <svcMenu>
      <version>1.0</version>
      <lang>en</lang>
      <lang>fr</lang>
      <objURI>urn:ietf:params:xml:ns:obj1</objURI>
      <objURI>urn:ietf:params:xml:ns:obj2</objURI>
      <objURI>urn:ietf:params:xml:ns:obj3</objURI>
      <svcExtension>
        <extURI>http://custom/obj1ext-1.0</extURI>
      </svcExtension>
    </svcMenu>
    <dcp>
      <access><all/></access>
      <statement>
        <purpose><admin/><prov/></purpose>
        <recipient><ours/><public/></recipient>
        <retention><stated/></retention>
      </statement>
    </dcp>
  </greeting>
</epp>
```

# Example

- We want to register two domains with the BILL registry:
  - where-is.bill (flag.ep.net, dot.ep.net)
  - platypus.bill (duck.platypus.bill, bill.platypus.bill)
- Let's see what host and domain transactions would be required

# where-is.bill – hosts

- We need to add the host objects to the registry before we can add a domain object which refers to them
- So, let's add host objects for flag.ep.net and dot.ep.net

# Create flag.ep.net

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
  epp-1.0.xsd">
  <command>
    <create>
      <host:create
        xmlns:host="urn:ietf:params:xml:ns:host-1.0"
        xsi:schemaLocation="urn:ietf:params:xml:ns:host-1.0
        host-1.0.xsd">
        <host:name>flag.ep.net</host:name>
        <host:addr ip="v4">198.32.4.13</host:addr>
        <host:addr ip="v4">61.8.9.254</host:addr>
        <host:addr ip="v6">3ffe:805::2d0:b7ff:fee8:c4d9</host:addr>
      </host:create>
    </create>
    <clTRID>ABC-12345</clTRID>
  </command>
</epp>
```



# Create dot.ep.net

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
  epp-1.0.xsd">
  <command>
    <create>
      <host:create
        xmlns:host="urn:ietf:params:xml:ns:host-1.0"
        xsi:schemaLocation="urn:ietf:params:xml:ns:host-1.0
        host-1.0.xsd">
        <host:name>flag.ep.net</host:name>
        <host:addr ip="v4">198.32.2.10</host:addr>
        <host:addr ip="v4">61.8.9.254</host:addr>
        <host:addr ip="v6">3ffe:0:1:0:230:48ff:fe22:6a29</host:addr>
        <host:addr ip="v6">2001:478:6:0:230:48ff:fe22:6a29</host:addr>
      </host:create>
    </create>
    <clTRID>ABC-12345</clTRID>
  </command>
</epp>
```

# Create where-is.bill

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
  epp-1.0.xsd">
  <command>
    <create>
      <domain:create
        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0"
        xsi:schemaLocation="urn:ietf:params:xml:ns:domain-1.0
        domain-1.0.xsd">
        <domain:name>where-is.bill</domain:name>
        <domain:period unit="y">2</domain:period>
        <domain:ns>flag.ep.net</domain:ns>
        <domain:ns>dot.ep.net</domain:ns>
        <domain:registrant>bm1234</domain:registrant>
        <domain:contact type="admin">bm1234</domain:contact>
        <domain:contact type="tech">bm1234</domain:contact>
        <domain:authInfo>
          <domain:pw>!!ferrets!!</domain:pw>
        </domain:authInfo>
      </domain:create>
    </create>
    <clTRID>ABC-12345</clTRID>
  </command>
</epp>
```

# platypus.bill – hosts

- Nameservers for platypus.bill are duck.platypus.bill, and bill.platypus.bill
- We are not allowed to add those hosts until their superordinate domain has been added
- The superordinate domain is platypus.bill
- We need the hosts to exist before the domain can be properly created
- Help!

# Create platypus.bill

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
  epp-1.0.xsd">
  <command>
    <create>
      <domain:create
        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0"
        xsi:schemaLocation="urn:ietf:params:xml:ns:domain-1.0
        domain-1.0.xsd">
        <domain:name>platypus.bill</domain:name>
        <domain:period unit="y">2</domain:period>
        <domain:registrant>bm1234</domain:registrant>
        <domain:contact type="admin">bm1234</domain:contact>
        <domain:contact type="tech">bm1234</domain:contact>
        <domain:authInfo>
          <domain:pw>!!ferrets!!</domain:pw>
        </domain:authInfo>
      </domain:create>
    </create>
    <clTRID>ABC-12345</clTRID>
  </command>
</epp>
```

# Add duck.platypus.bill

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
  epp-1.0.xsd">
  <command>
    <create>
      <host:create
        xmlns:host="urn:ietf:params:xml:ns:host-1.0"
        xsi:schemaLocation="urn:ietf:params:xml:ns:host-1.0
        host-1.0.xsd">
        <host:name>duck.platypus.bill</host:name>
        <host:addr ip="v4">192.168.78.1</host:addr>
      </host:create>
    </create>
    <clTRID>ABC-12345</clTRID>
  </command>
</epp>
```

# Add bill.platypus.bill

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
  epp-1.0.xsd">
  <command>
    <create>
      <host:create
        xmlns:host="urn:ietf:params:xml:ns:host-1.0"
        xsi:schemaLocation="urn:ietf:params:xml:ns:host-1.0
        host-1.0.xsd">
        <host:name>duck.platypus.bill</host:name>
        <host:addr ip="v4">192.168.87.1</host:addr>
      </host:create>
    </create>
    <clTRID>ABC-12345</clTRID>
  </command>
</epp>
```

# Update platypus.bill

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
  epp-1.0.xsd">
  <command>
    <update>
      <domain:update
        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0"
        xsi:schemaLocation="urn:ietf:params:xml:ns:domain-1.0
        domain-1.0.xsd">
        <domain:name>platypus.bill</domain:name>
        <domain:add>
          <domain:ns>duck.platypus.bill</domain:ns>
          <domain:ns>bill.platypus.bill</domain:ns>
        </domain:add>
      </domain:update>
    </update>
    <c1TRID>ABC-12345</c1TRID>
  </command>
</epp>
```

# Additional!

- Attempting to add a host object that already exists will result in an error
- Registrars need to do a structured series of <host:check>, <host:create>, <domain:check>, <domain:create> and <domain:update> commands in order to create a domain
- It has been said that this is a little cumbersome



# EPP Status

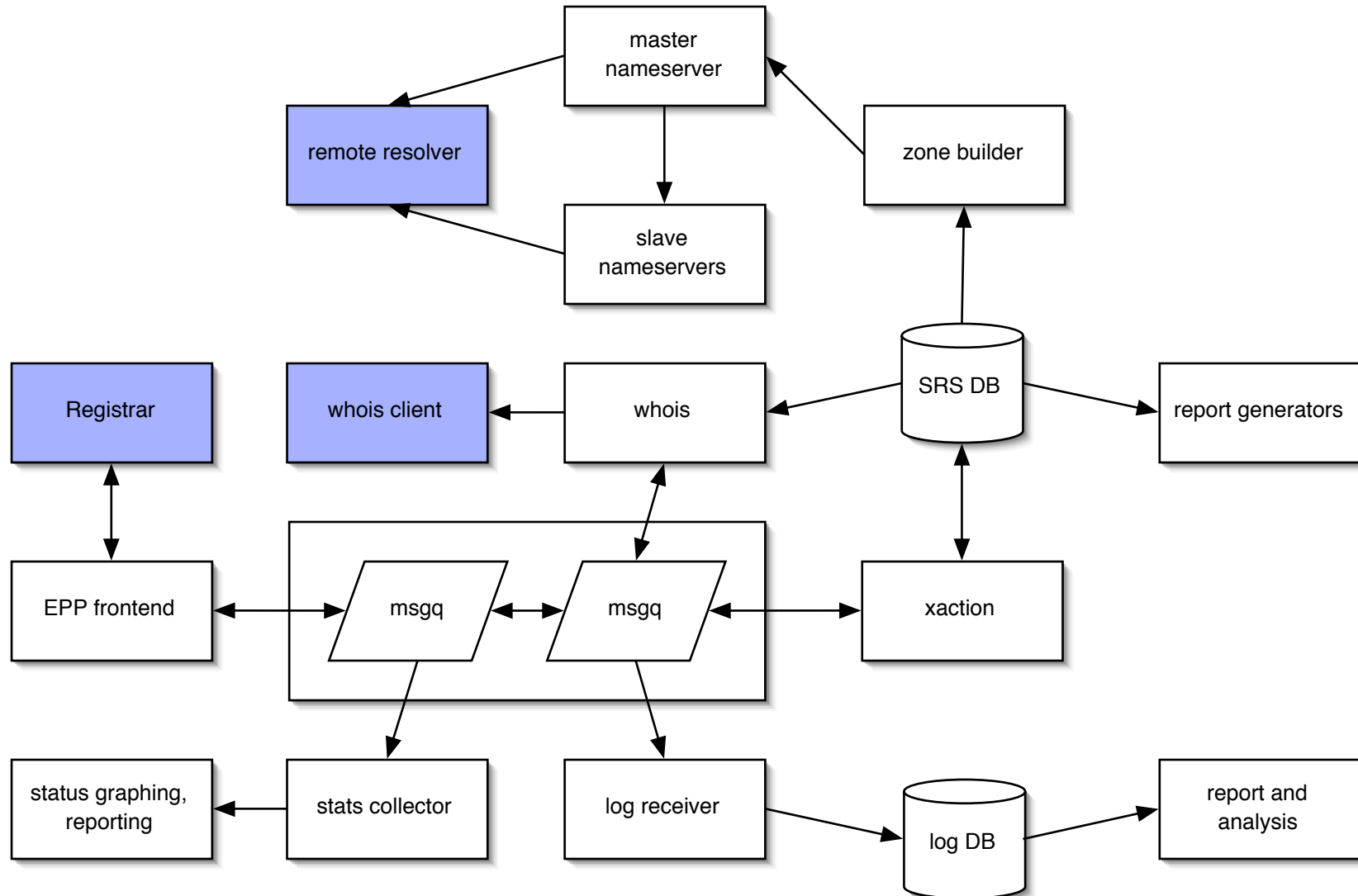
- A version of the EPP specification is currently before the IESG, waiting to be accepted as a draft standard
- Currently held up on a couple of issues, mainly relating to privacy
- Once Ed has beaten some consensus out of the whining throng in the provreg wg, implementation and interop can proceed

ISC OpenReg

# OpenReg Goals

- Standards Compliance
- High Performance
  - originally implemented to run ORG
- Scalable, Reliable
  - eliminate single points of failure
- Free Software (BSD-like licence)

# Components



# Component Distribution

- Multiple xaction components
- Multiple epp-frontend components
- Distribution of components across multiple machines
  - joined by a mesh of msgq processes
- Fail-over betwen xaction components

# Logging, Statistics Gathering

- All components generate log messages and performance information
- Performance and logging data are sent over the msgbus
- Picked up by one or more statistics servers and log receivers

# Status

- First public release occurred earlier this year
- Latest release is OpenReg 1.0.2
- There are developers working with OpenReg in real registries

# What Works?

- Session control (over TCP transport)
- Basic operations on domain, host and contact objects (create, delete, update, info, check)
- Domain renew
- Zone file creation
- Logging, statistics generation and collection
- whois



# What Doesn't Work?

- Multi-host msgq (the API supports it, but the msgq daemons don't)
- Redundant xaction components
- Domain transfer

# Future

- We plan to continue to dedicate developer resource to the project
- We are actively seeking funding, so that we can provide long-term development support to the project
- We will continue to give Ed a hard time

# Practice!

- Download OpenReg 1.0.2
- Unpack the tarball
  - openreg-1.0.2/README
  - openreg-1.0.2/REQUIRED-SOFTWARE
  - openreg-1.0.2/INSTALL