The Problem

- How to apply profile-specific verification rules that:
  - Keeps verification data local
  - Maintains the Registrar and Registry relationship
  - Has one auditable verification
  - Supports confirmation of verification of any object and any operation
  - Separates verification interface from the Registry interface
  - Is extensible for multiple profiles and diverse rules
- The Registry Verification Framework solves this problem
Verification World

Registry .EXAMPLE

Profile

Registry

Registrar

Registrar to Registry
Framework Basics

- Verification Service Provider (VSP)
  - VSPs perform the Verification required for a Profile
  - Registrars Interface with VSP for Verification
  - VSP Generates Signed Verification Codes
  - Registry has the VSP Trust Anchor to Verify VSP Generated Codes
Framework Basics Cont.

- Verification Codes
  - Compliant with draft-gould-eppext-verificationcode
  - Globally Unique Token Value Per Verification
    - Format: vsp-id "-" verification-id
  - Typed to Identify Verification Performed
    - Example: “domain”
  - BASE64 encoded XML Signature

- Registry Profile
  - Defines the Policy for the Profile
Framework Components and Relationships

- **Foo**
  - Bar
    - Bar-R1
  - Foo-R1
  - Foo-R2

- **Registry**
  - Baz
    - Baz-R1
    - Baz-R2

- **Registrar to Registry**
- **Registrar to VSP**
- **Profile**
- **Registry**
- **VSP**
Framework Components and Relationships

Registry

Profile

Registrar

VSP

Registrar to Registry

Registrar to VSP

Foo

Registry .EXAMPLE

Bar

Baz

Foo-V1

Foo-R1

Bar-R1

Foo-R2

Baz-R1

Baz-R2
Framework Components and Relationships

- **Foo**
  - Foo-V1
  - Foo-R1
  - Foo-R2

- **Bar**
  - Bar-R1
  - Bar-V1

- **Registry .EXAMPLE**

- **Baz**
  - Baz-R1
  - Baz-R2

Legend:
- Red: Profile
- Blue: Registrar
- Purple: VSP
- Green arrows: Registrar to Registry
- Purple arrow: Registrar to VSP
Framework Components and Relationships

Diagram showing relationships between different components:
- Foo to Bar
- Foo-V1 to Bar-R1
- Foo-R1 to Bar-R1
- Foo-R2 to Bar-V1
- Baz to Registry
- Baz-R1 to Baz-R1
- Baz-R2 to Baz-R2

Symbols:
- Profile
- Registry
- Registrar
- VSP
- Registrar to Registry
- Registrar to VSP
Framework Components and Relationships

- **Foo**
  - Foo-V1
  - Foo-R1
  - Foo-R2

- **Bar**
  - Bar-R1
  - Bar-V1

- **Registry .EXAMPLE**

- **Baz**
  - Baz-R1
  - Baz-V1
  - Baz-R2

Legend:
- Profile
- Registry
- Registrar
- VSP
- Registrar to Registry
- Registrar to VSP
Registry Profile Model

Registry Profile

VSP

Code Type

Registrar

1..n

1..n

0..n
Verification Profiles for Verification World

- VSP
  - Foo-V1
  - Foo-R2
- Code Type "registrant"
  - Foo
  - Registrar
    - Foo-R1
    - Foo-R2
    - Foobar-R1
Verification Profiles for Verification World

- **Foo**
  - VSP: Foo-V1, Foo-R2
  - Code Type: "registrant"
    - Registrar:
      - Foo-R1
      - Foo-R2
      - Foobar-R1

- **Bar**
  - VSP: Bar-V1
  - Code Type: "domain"
    - Registrar: Bar-R1
Verification Profiles for Verification World

Foo
- VSP Foo-V1
- Code Type “registrant”
  - Registrar Foo-R1
  - Foo-R2
  - Foobar-R1

Bar
- VSP Bar-V1
- Code Type “domain”
  - Registrar Bar-R1

Baz
- VSP Baz-V1
- Code Type “registrant” “domain”
  - Registrar Baz-R1
  - Baz-R2
Problem Checklist

☑ Keeps verification data local
☑ Maintains the Registrar and Registry relationship
☑ Has one auditable verification
☑ Supports confirmation of verification of any object and any operation
☑ Separates verification interface from the Registry interface
☑ Is extensible for multiple profiles and diverse rules
Conclusion

The Registry Verification Framework defines a flexible framework for implementing verification requirements of verification profiles while maintaining the Registrar and Registry relationship and ensuring that the verification data stays within the scope of the profile.