

Network Working Group  
Internet-Draft  
Intended status: Best Current Practice  
Expires: February 6, 2020

J. Gould  
R. Wilhelm  
VeriSign, Inc.  
August 5, 2019

Extensible Provisioning Protocol (EPP) Secure Authorization Information  
for Transfer

[draft-gould-regext-secure-authinfo-transfer-02](#)

Abstract

The Extensible Provisioning Protocol (EPP), in [RFC 5730](#), defines the use of authorization information to authorize a transfer. The authorization information is object-specific and has been defined in the EPP Domain Name Mapping, in [RFC 5731](#), and the EPP Contact Mapping, in [RFC 5733](#), as password-based authorization information. Other authorization mechanisms can be used, but in practice the password-based authorization information has been used at the time of object create, managed with the object update, and used to authorize an object transfer request. What has not been fully considered is the security of the authorization information that includes the complexity of the authorization information, the time-to-live (TTL) of the authorization information, and where and how the authorization information is stored. This document defines an operational practice, using the EPP RFCs, that leverages the use of strong random authorization information values that are short-lived, that are not stored by the client, and that are stored using a cryptographic hash by the server to provide for secure authorization information used for transfers.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 6, 2020.

## Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">2</a>
<a href="#">1.1.</a>	Conventions Used in This Document . . . . .	<a href="#">4</a>
<a href="#">2.</a>	Registrant, Registrar, Registry . . . . .	<a href="#">4</a>
<a href="#">3.</a>	Secure Authorization Information . . . . .	<a href="#">5</a>
<a href="#">3.1.</a>	Secure Random Authorization Information . . . . .	<a href="#">6</a>
<a href="#">3.2.</a>	Authorization Information Time-To-Live (TTL) . . . . .	<a href="#">7</a>
<a href="#">3.3.</a>	Authorization Information Storage and Transport . . . . .	<a href="#">7</a>
<a href="#">4.</a>	Create, Transfer, and Secure Authorization Information . . . . .	<a href="#">7</a>
<a href="#">4.1.</a>	Create Command . . . . .	<a href="#">8</a>
<a href="#">4.2.</a>	Update Command . . . . .	<a href="#">10</a>
<a href="#">4.3.</a>	Info Command and Response . . . . .	<a href="#">14</a>
<a href="#">4.4.</a>	Transfer Request Command . . . . .	<a href="#">14</a>
<a href="#">5.</a>	Implementation Status . . . . .	<a href="#">15</a>
<a href="#">5.1.</a>	Verisign EPP SDK . . . . .	<a href="#">16</a>
<a href="#">5.2.</a>	RegistryEngine EPP Service . . . . .	<a href="#">16</a>
<a href="#">6.</a>	Security Considerations . . . . .	<a href="#">17</a>
<a href="#">7.</a>	Acknowledgements . . . . .	<a href="#">17</a>
<a href="#">8.</a>	References . . . . .	<a href="#">17</a>
<a href="#">8.1.</a>	Normative References . . . . .	<a href="#">17</a>
<a href="#">8.2.</a>	URIs . . . . .	<a href="#">18</a>
<a href="#">Appendix A.</a>	Change History . . . . .	<a href="#">18</a>
<a href="#">A.1.</a>	Change from 00 to 01 . . . . .	<a href="#">18</a>
<a href="#">A.2.</a>	Change from 01 to 02 . . . . .	<a href="#">18</a>
	Authors' Addresses . . . . .	<a href="#">18</a>

## [1.](#) Introduction

The Extensible Provisioning Protocol (EPP), in [[RFC5730](#)], defines the use of authorization information to authorize a transfer. The authorization information is object-specific and has been defined in the EPP Domain Name Mapping, in [[RFC5731](#)], and the EPP Contact

Gould & Wilhelm

Expires February 6, 2020

[Page 2]

Mapping, in [[RFC5733](#)], as password-based authorization information. Other authorization mechanisms can be used, but in practice the password-based authorization information has been used at the time of object create, managed with the object update, and used to authorize an object transfer request. What has not been considered is the security of the authorization information that includes the complexity of the authorization information, the time-to-live (TTL) of the authorization information, and where and how the authorization information is stored. This document defines an operational practice, using the EPP RFCs, that leverages the use of strong, random authorization information values that are short-lived, that are not stored by the client, and that are stored by the server using a cryptographic hash to provide, for secure authorization information used for transfers. This operational practice can be used to support transfers of any EPP object, where the domain name object defined in [[RFC5731](#)] is used in this document for illustration purposes.

The overall goal is to have strong, random authorization information values, that are short-lived, and that are either not stored or stored as a cryptographic hash values by the non-responsible parties. In a registrant, registrar, and registry model, the registrant registers the object through the registrar to the registry. The registrant is the responsible party and the registrar and the registry are the non-responsible parties. EPP is a protocol between the registrar and the registry, where the registrar is referred to as the client and the registry is referred to as the server. The following are the elements of the operational practice and how the existing features of the EPP RFCs can be leveraged to satisfy them:

"Strong Random Authorization Information": The EPP RFCs define the password-based authorization information value using an XML schema "normalizedString" type, so they don't restrict what can be used in any way. This operational practice defines the recommended mechanism for creating a strong random authorization value, that would be generated by the client.

"Short-Lived Authorization Information": The EPP RFCs don't explicitly support short-lived authorization information or a time-to-live (TTL) for authorization information, but there are EPP RFC features that can be leveraged to support short-lived authorization information. If authorization information is set only when there is a transfer in process, the server needs to support empty authorization information on create, support setting and unsetting authorization information, and support automatically unsetting the authorization information upon a successful transfer. All of these features can be supported by the EPP RFCs.

"Storing Authorization Information Securely": The EPP RFCs don't specify where and how the authorization information is stored in



the client or the server, so there are no restrictions to define an operational practice for storing the authorization information securely. The operational practice will not require the client to store the authorization information and will require the server to store the authorization information using a cryptographic hash. Returning the authorization information set in an EPP info response will not be supported.

### **1.1. Conventions Used in This Document**

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#) [[RFC2119](#)].

XML is case sensitive. Unless stated otherwise, XML specifications and examples provided in this document MUST be interpreted in the character case presented in order to develop a conforming implementation.

In examples, "C:" represents lines returned by a protocol client. Indentation and white space in examples are provided only to illustrate element relationships and are not a REQUIRED feature of this protocol.

The examples reference XML namespace prefixes that are used for the associated XML namespaces. Implementations MUST NOT depend on the example XML namespaces and instead employ a proper namespace-aware XML parser and serializer to interpret and output the XML documents. The example namespace prefixes used and their associated XML namespaces include:

```
"domain": urn:ietf:params:xml:ns:domain-1.0
"contact": urn:ietf:params:xml:ns:contact-1.0
```

## **2. Registrant, Registrar, Registry**

The EPP RFCs refer to client and server, but when it comes to transfers, there are three actors that are involved. This document will refer to the actors as registrant, registrar, and registry. [[RFC8499](#)] defines these terms formally for the Domain Name System (DNS). The terms are further described below to cover their roles as actors of using the authorization information in the transfer process of any object in the registry, such as a domain name or a contact:

"registrant": [[RFC8499](#)] defines the registrant as "an individual or organization on whose behalf a name in a zone is registered by the registry". The registrant can be the owner of any object in the registry, such as a domain name or a contact. The registrant



interfaces with the registrar for provisioning the objects. A transfer is coordinated by the registrant to transfer the sponsorship of the object from one registrar to another. The authorization information is meant to authenticate the registrant as the owner of the object to the non-sponsoring registrar and to authorize the transfer.

"registrar": [\[RFC8499\]](#) defines the registrar as "a service provider that acts as a go-between for registrants and registries". The registrar interfaces with the registrant for the provisioning of objects, such as domain names and contacts, and with the registries to satisfy the registrant's provisioning requests. A registrar may directly interface with the registrant or may indirectly interface with the registrant, typically through one or more resellers. Implementing a transfer using secure authorization information extends through the registrar's reseller channel up to the direct interface with the registrant. The registrar's interface with the registries uses EPP. The registrar's interface with its reseller channel or the registrant is registrar-specific. In the EPP RFCs, the registrar is referred to as the "client", since EPP is the protocol used between the registrar and the registry. The sponsoring registrar is the authorized registrar to manage objects on behalf of the registrant. A non-sponsoring registrar is not authorized to manage objects on behalf of the registrant. A transfer of an object's sponsorship is from one registrar, referred to as the losing registrar, to another registrar, referred to as the gaining registrar.

"registry": [\[RFC8499\]](#) defines the registry as "the administrative operation of a zone that allows registration of names within the zone". The registry typically interfaces with the registrars over EPP and generally does not interact directly with the registrant. In the EPP RFCs, the registry is referred to as the "server", since EPP is the protocol used between the registrar and the registry. The registry has a record of the sponsoring registrar for each object and provides the mechanism (over EPP) to coordinate a transfer of an object's sponsorship between registrars.

### **3. Secure Authorization Information**

The authorization information in the EPP RFCs ([\[RFC5731\]](#) and [\[RFC5733\]](#)) that support transfer use password-based authorization information. Other EPP objects that support password-based authorization information for transfer can use the Secure Authorization Information defined in this document. For the authorization information to be secure it must be a strong random value and must have a short time-to-live (TTL). The security of the authorization information is defined in the following sections.



### **3.1. Secure Random Authorization Information**

For authorization information to be secure, it MUST be generated using a secure random value. The authorization information is treated as a password, where according to [RFC4086] a high-security password must have at least 49 bits of randomness or entropy. The required length  $L$  of a password, rounded up to the largest whole number, is based on the set of characters  $N$  and the desired entropy  $H$ , in the equation  $L = \text{ROUNDUP}(H / \log_2 N)$ . With a target entropy of 49, the required length can be calculated after deciding on the set of characters that will be randomized. The following are a set of possible character sets and the calculation of the required length.

Calculation of the required length with 49 bits of entropy and with the set of all printable ASCII characters except space (0x20), which consists of the 94 characters 0x21-0x7E.

$$\text{ROUNDUP}(49 / \log_2 94) = \text{ROUNDUP}(49 / 6.55) = \text{ROUNDUP}(7.48) = 8$$

Calculation of the required length with 49 bits of entropy and with the set of case-insensitive alphanumeric characters, which consists of 36 characters (a-z A-Z 0-9).

$$\text{ROUNDUP}(49 / \log_2 36) = \text{ROUNDUP}(49 / 5.17) = \text{ROUNDUP}(9.48) = 10$$

Considering the age of [RFC4086], the evolution of security practices, and that the authorization information is a machine-generated value, the recommendation is to use at least 128 bits of entropy. The lengths are recalculated below using 128 bits of entropy.

Calculation of the required length with 128 bits of entropy and with the set of all printable ASCII characters except space (0x20), which consists of the 94 characters 0x21-0x7E.

$$\text{ROUNDUP}(128 / \log_2 94) = \text{ROUNDUP}(128 / 6.55) = \text{ROUNDUP}(19.54) = 20$$

Calculation of the required length with 128 bits of entropy and with the set of case insensitive alphanumeric characters, which consists of 36 characters (a-z A-Z 0-9).

$$\text{ROUNDUP}(128 / \log_2 36) = \text{ROUNDUP}(128 / 5.17) = \text{ROUNDUP}(24.76) = 25$$

The strength of the random authorization information is dependent on the actual entropy of the underlying random number generator. For the random number generator, the practices defined in [RFC4086] and [section 4.7.1](#) of the NIST Federal Information Processing Standards (FIPS) Publication 140-2 [1] SHOULD be followed to produce random



values that will be resistant to attack. A random number generator (RNG) is preferable over the use of a pseudorandom number generator (PRNG) to reduce the predictability of the authorization information. The more predictable the random number generator is, the lower the true entropy, and the longer the required length for the authorization information.

### **3.2. Authorization Information Time-To-Live (TTL)**

The authorization information SHOULD only be set when there is a transfer in process. This implies that the authorization information has a Time-To-Live (TTL) by which the authorization information is cleared when the TTL expires. The EPP RFCs have no definition of TTL, but since the server supports the setting and unsetting of the authorization information by the sponsoring registrar, then the sponsoring registrar can apply a TTL based on client policy. The TTL client policy may be based on proprietary registrar-specific criteria which provides for a transfer-specific TTL tuned for the particular circumstances of the transaction. The sponsoring registrar will be aware of the TTL and the sponsoring registrar MUST inform the registrant of the TTL when the authorization information is provided to the registrant.

### **3.3. Authorization Information Storage and Transport**

To protect the disclosure of the authorization information, the authorization information MUST be stored by the registry using a strong one-way cryptographic hash, MUST NOT be stored by the losing registrar, and MUST only be stored by the gaining registrar as a "transient" value in support of the transfer process. The plain text version of the authorization information MUST NOT be written to any logs by the registrar or the registry. All communication that includes the authorization information MUST be over an encrypted channel, such as [\[RFC5734\]](#) for EPP. The registrar's interface for communicating the authorization information with the registrant MUST be over an authenticated and encrypted channel.

## **4. Create, Transfer, and Secure Authorization Information**

To make the transfer process secure using secure authorization information, as defined in [Section 3](#), the client and server need to implement steps where the authorization information is set only when a transfer is actively in process and ensure that the authorization information is stored securely and transported only over secure channels. The steps in management of the authorization information for transfers include:



1. Registrant requests to register the object with the registrar. Registrar sends the create command, with empty authorization information, to the registry, as defined in [Section 4.1](#).
2. Registrant requests from the losing registrar the authorization information to provide to the gaining registrar.
3. Losing registrar generates a secure random authorization information value, sends it to the registry as defined in [Section 4.2](#), and provides it to the registrant.
4. Registrant provides the authorization information value to the gaining registrar.
5. Gaining registrar optionally verifies the authorization information with the info command to the registry, as defined in [Section 4.3](#).
6. Gaining registrar sends the transfer request with the authorization information to the registry, as defined in [Section 4.4](#).
7. If the transfer successfully completes, the registry automatically unsets the authorization information; otherwise the losing registrar unsets the authorization information when the TTL expires, as defined in [Section 4.2](#).

The following sections outline the practices of the EPP commands and responses between the registrar and the registry that supports secure authorization information for transfer.

#### **[4.1](#). Create Command**

For a Create Command, the registry MUST allow for the passing of an empty authorization information and MAY disallow for the passing of a non-empty authorization information. By having an empty authorization information on create, the object is initially not in the transfer process. Any EPP object extension that supports setting the authorization information with a "eppcom:pwAuthInfoType" element, can have an empty authorization information passed, such as [\[RFC5731\]](#) and [\[RFC5733\]](#).



Example of passing empty authorization information in an [\[RFC5731\]](#) domain name create command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <create>
C:      <domain:create
C:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:          <domain:name>example.com</domain:name>
C:          <domain:authInfo>
C:            <domain:pw/>
C:          </domain:authInfo>
C:        </domain:create>
C:      </create>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```



Example of passing empty authorization information in an [\[RFC5733\]](#) contact create command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <create>
C:      <contact:create
C:        xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:          <contact:id>sh8013</contact:id>
C:          <contact:postalInfo type="int">
C:            <contact:name>John Doe</contact:name>
C:            <contact:org>Example Inc.</contact:org>
C:            <contact:addr>
C:              <contact:street>123 Example Dr.</contact:street>
C:              <contact:street>Suite 100</contact:street>
C:              <contact:city>Dulles</contact:city>
C:              <contact:sp>VA</contact:sp>
C:              <contact:pc>20166-6503</contact:pc>
C:              <contact:cc>US</contact:cc>
C:            </contact:addr>
C:          </contact:postalInfo>
C:          <contact:voice x="1234">+1.7035555555</contact:voice>
C:          <contact:fax>+1.7035555556</contact:fax>
C:          <contact:email>jdoe@example.com</contact:email>
C:          <contact:authInfo>
C:            <contact:pw/>
C:          </contact:authInfo>
C:          <contact:disclose flag="0">
C:            <contact:voice/>
C:            <contact:email/>
C:          </contact:disclose>
C:        </contact:create>
C:      </create>
C:      <clTRID>ABC-12345</clTRID>
C:    </command>
C:</epp>
```

#### [4.2.](#) Update Command

For an Update Command, the registry MUST allow for the setting and unsetting of the authorization information. The registrar sets the authorization information by first generating a strong, random authorization information value, based on [Section 3.1](#), and setting it in the registry in the update command. The registry SHOULD validate the randomness of the authorization information based on the length and character set required by the registry. For example, a registry that requires 20 random printable ASCII characters except space



(0x20), should validate that the authorization information contains at least one upper case alpha character, one lower case alpha character, and one non-alpha numeric character. If the authorization information fails the randomness validation, the registry MUST return an EPP error result code of 2202.

Often the registrar has the "clientTransferProhibited" status set, so to start the transfer process, the "clientTransferProhibited" status needs to be removed, and the strong, random authorization information value needs to be set. The registrar MUST define a time-to-live (TTL), as defined in [Section 3.2](#), where if the TTL expires the registrar will unset the authorization information.

Example of removing the "clientTransferProhibited" status and setting the authorization information in an [\[RFC5731\]](#) domain name update command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <update>
C:      <domain:update
C:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:        <domain:name>example.com</domain:name>
C:        <domain:rem>
C:          <domain:status s="clientTransferProhibited"/>
C:        </domain:rem>
C:        <domain:chg>
C:          <domain:authInfo>
C:            <domain:pw>LuQ7Bu@w9?%+_HK3cayg$55$LSft3MPP
C:          </domain:pw>
C:          </domain:authInfo>
C:        </domain:chg>
C:      </domain:update>
C:    </update>
C:    <clTRID>ABC-12345-XYZ</clTRID>
C:  </command>
C:</epp>
```

When the registrar-defined TTL expires, the sponsoring registrar cancels the transfer process by unsetting the authorization information value and may add back statuses like the "clientTransferProhibited" status. Any EPP object extension that supports setting the authorization information with a "eppcom:pwAuthInfoType" element, can have an empty authorization information passed, such as [\[RFC5731\]](#) and [\[RFC5733\]](#). Setting an empty authorization information unsets the value. [\[RFC5731\]](#) supports an explicit mechanism of unsetting the authorization information, by



passing the <domain:null> authorization information value. The registry MUST support unsetting the authorization information by accepting an empty authorization information value and accepting an explicit unset element if it is supported by the object extension.

Example of unsetting the authorization information explicitly in an [\[RFC5731\]](#) domain name update command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <update>
C:      <domain:update
C:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:        <domain:name>example.com</domain:name>
C:        <domain:add>
C:          <domain:status s="clientTransferProhibited"/>
C:        </domain:add>
C:        <domain:chg>
C:          <domain:authInfo>
C:            <domain:null/>
C:          </domain:authInfo>
C:        </domain:chg>
C:      </domain:update>
C:    </update>
C:    <clTRID>ABC-12345-XYZ</clTRID>
C:  </command>
C:</epp>
```



Example of unsetting the authorization information with an empty authorization information in an [[RFC5731](#)] domain name update command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <update>
C:      <domain:update
C:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:        <domain:name>example.com</domain:name>
C:        <domain:add>
C:          <domain:status s="clientTransferProhibited"/>
C:        </domain:add>
C:        <domain:chg>
C:          <domain:authInfo>
C:            <domain:pw/>
C:          </domain:authInfo>
C:        </domain:chg>
C:      </domain:update>
C:    </update>
C:    <clTRID>ABC-12345-XYZ</clTRID>
C:  </command>
C:</epp>
```

Example of unsetting the authorization information with an empty authorization information in an [[RFC5733](#)] contact update command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <update>
C:      <contact:update
C:        xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:        <contact:id>sh8013</contact:id>
C:        <contact:chg>
C:          <contact:authInfo>
C:            <contact:pw/>
C:          </contact:authInfo>
C:        </contact:chg>
C:      </contact:update>
C:    </update>
C:    <clTRID>ABC-12345-XYZ</clTRID>
C:  </command>
C:</epp>
```



### 4.3. Info Command and Response

For an Info Command, the registry MUST allow for the passing of a non-empty authorization information for verification. The gaining registrar can pre-verify the authorization information provided by the registrant prior to submitting the transfer request with the use of the Info Command. The registry compares the hash of the passed authorization information with the hashed authorization information value stored for the object. When the authorization information is not set or the passed authorization information does not match the previously set value, the registry MUST return an EPP error result code of 2202 [[RFC5730](#)].

Example of passing a non-empty authorization information in an [[RFC5731](#)] domain name info command to verify the authorization information value.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <info>
C:      <domain:info
C:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:          <domain:name>example.com</domain:name>
C:          <domain:authInfo>
C:            <domain:pw>LuQ7Bu@w9?%+_HK3cayg$55$LSft3MPP
C:          </domain:pw>
C:        </domain:authInfo>
C:      </domain:info>
C:    </info>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

The Info Response in object extensions, such as [[RFC5731](#)] and [[RFC5733](#)], MUST NOT include the optional authorization information element. The authorization information is stored as a hash in the registry, so returning the plain text authorization information is not possible. The registry MUST NOT return any indication of whether the authorization information is set or unset by not returning the authorization information element in the response.

### 4.4. Transfer Request Command

For a Transfer Request Command, the registry MUST allow for the passing of a non-empty authorization information to authorize a transfer. The registry compares the hash of the passed authorization information with the hashed authorization information value stored



for the object. When the authorization information is not set or the passed authorization information does not match the previously set value, the registry MUST return an EPP error result code of 2202 [RFC5730]. Whether the transfer occurs immediately or is pending is up to server policy. When the transfer occurs immediately, the registry MUST return the EPP success result code of 1000 and when the transfer is pending, the registry MUST return the EPP success result code of 1001. The losing registrar MUST be informed of a successful transfer request using an EPP poll message.

Example of passing a non-empty authorization information in an [RFC5731] domain name transfer request command to authorize the transfer.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <transfer op="request">
C:      <domain:transfer
C:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:        <domain:name>example1.com</domain:name>
C:        <domain:authInfo>
C:          <domain:pw>LuQ7Bu@w9?%+_HK3cayg$55$LSft3MPP
C:          </domain:pw>
C:        </domain:authInfo>
C:      </domain:transfer>
C:    </transfer>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

Upon successful completion of the transfer, the registry MUST automatically unset the authorization information. If the transfer does not complete within the time-to-live (TTL) (Section 3.2), the registrar MUST unset the authorization information as defined in Section 4.2.

## 5. Implementation Status

Note to RFC Editor: Please remove this section and the reference to RFC 7942 [RFC7942] before publication.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in RFC 7942 [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual



implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC 7942](#) [[RFC7942](#)], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

### **[5.1.](#) Verisign EPP SDK**

Organization: Verisign Inc.

Name: Verisign EPP SDK

Description: The Verisign EPP SDK includes both a full client implementation and a full server stub implementation of [draft-gould-regext-secure-authinfo-transfer](#).

Level of maturity: Development

Coverage: All aspects of the protocol are implemented.

Licensing: GNU Lesser General Public License

Contact: [jgould@verisign.com](mailto:jgould@verisign.com)

URL: [https://www.verisign.com/en\\_US/channel-resources/domain-registry-products/epp-sdks](https://www.verisign.com/en_US/channel-resources/domain-registry-products/epp-sdks)

### **[5.2.](#) RegistryEngine EPP Service**

Organization: CentralNic

Name: RegistryEngine EPP Service

Description: Generic high-volume EPP service for gTLDs, ccTLDs and SLDs

Level of maturity: Deployed in CentralNic's production environment as well as two other gTLD registry systems, and two ccTLD registry systems.



Coverage: Auhtorization Information is "write only" in that the registrars can set the Auhtorization Information, but not get the Auhtorization Information in the Info Response.

Licensing: Proprietary In-House software

Contact: epp@centralnic.com

URL: <https://www.centralnic.com>

## **6. Security Considerations**

TBD

## **7. Acknowledgements**

The authors wish to thank the following persons for their feedback and suggestions:

- o Scott Hollenbeck
- o Jody Kolker
- o Patrick Mevzek
- o Matthew Pozun
- o Srikanth Veeramachaneni

## **8. References**

### **8.1. Normative References**

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", [BCP 106](#), [RFC 4086](#), DOI 10.17487/RFC4086, June 2005, <<https://www.rfc-editor.org/info/rfc4086>>.
- [RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, [RFC 5730](#), DOI 10.17487/RFC5730, August 2009, <<https://www.rfc-editor.org/info/rfc5730>>.
- [RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, [RFC 5731](#), DOI 10.17487/RFC5731, August 2009, <<https://www.rfc-editor.org/info/rfc5731>>.



- [RFC5733] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Contact Mapping", STD 69, [RFC 5733](#), DOI 10.17487/RFC5733, August 2009, <<https://www.rfc-editor.org/info/rfc5733>>.
- [RFC5734] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Transport over TCP", STD 69, [RFC 5734](#), DOI 10.17487/RFC5734, August 2009, <<https://www.rfc-editor.org/info/rfc5734>>.
- [RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", [BCP 205](#), [RFC 7942](#), DOI 10.17487/RFC7942, July 2016, <<https://www.rfc-editor.org/info/rfc7942>>.
- [RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", [BCP 219](#), [RFC 8499](#), DOI 10.17487/RFC8499, January 2019, <<https://www.rfc-editor.org/info/rfc8499>>.

## **[8.2.](#) URIs**

- [1] <https://csrc.nist.gov/publications/detail/fips/140/2/final>

## **[Appendix A.](#) Change History**

### **[A.1.](#) Change from 00 to 01**

1. Filled in the "Implementation Status" section with the inclusion of the "Verisign EPP SDK" and "RegistryEngine EPP Service" implementations.
2. Made small wording corrections based on private feedback.
3. Added content to the "Acknowledgements" section.

### **[A.2.](#) Change from 01 to 02**

1. Revised the language used for the storage of the authorization information based on the feedback from Patrick Mevzek and Jody Kolker.

## Authors' Addresses

James Gould  
VeriSign, Inc.  
12061 Bluemont Way  
Reston, VA 20190  
US

Email: [jgould@verisign.com](mailto:jgould@verisign.com)  
URI: <http://www.verisign.com>



Richard Wilhelm  
VeriSign, Inc.  
12061 Bluemont Way  
Reston, VA 20190  
US

Email: [rwilhelm@verisign.com](mailto:rwilhelm@verisign.com)

URI: <http://www.verisign.com>