Workgroup: Network Working Group
Internet-Draft:
draft-ietf-regext-secure-authinfo-transfer-07
Published: 28 June 2021
Intended Status: Standards Track
Expires: 30 December 2021
Authors: J. Gould
VeriSign, Inc.
R. Wilhelm
VeriSign, Inc.

# Extensible Provisioning Protocol (EPP) Secure Authorization Information for Transfer

## Abstract

The Extensible Provisioning Protocol (EPP), in RFC 5730, defines the use of authorization information to authorize a transfer of an EPP object, such as a domain name, between clients that are referred to as registrars. Object-specific, password-based authorization information (see RFC 5731 and RFC 5733) is commonly used, but raises issues related to the security, complexity, storage, and lifetime of authentication information. This document defines an operational practice, using the EPP RFCs, that leverages the use of strong random authorization information values that are short-lived, not stored by the client, and stored by the server using a cryptographic hash that provides for secure authorization information that can safely be used for object 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 30 December 2021.

**Table of Contents**

## 1. Introduction

The Extensible Provisioning Protocol (EPP), in [RFC5730], defines the use of authorization information to authorize a transfer of an EPP object, such as a domain name, between clients that are referred to as registrars. The authorization information is object-specific and has been defined in the EPP Domain Name Mapping, in [RFC5731], and the EPP Contact 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.

The current/original lifecycle for authorization information involves long-term storage of encrypted (not hashed) passwords, which presents a significant latent risk of password compromise and is not consistent with current best practices. The mechanisms in this document provide a way to avoid long-term password storage entirely, and to only require the storage of hashed (not retrievable) passwords instead of encrypted passwords.

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 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. Elements of the practice may be used to support the secure use of the authorization information for purposes other than transfer, but any other purposes and the applicable elements are out-of-scope for this document.

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 substantial 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. All of these features are compatible with the EPP RFCs, though not mandatory to implement. In section 2.6 of [RFC5731] it states that authorization information is assigned when a domain object is created, which results in long-lived authorization information. This specification changes the nature of the authorization information to be short-lived. If authorization information is set only when there is a transfer in process, the server needs to support an empty authorization information value 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 require the client to not store the authorization information and will require the server to store the authorization information using a cryptographic hash, with at least a 256-bit hash function, such as SHA-256 [FIPS-180-4], and with a per-authorization information random salt, with at least 128 bits. 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", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

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 sent by a protocol client and "S:" represents lines returned by a protocol server. 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 types of 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. Signaling Client and Server Support

This document does not define new protocol but an operational practice using the existing EPP protocol, where the client and the server can signal support for the operational practice using a namespace URI in the login and greeting extension services. The namespace URI "urn:ietf:params:xml:ns:epp:secure-authinfo-transfer-1.0" is used to signal support for the operational practice. The client includes the namespace URI in an <svcExtension> <extURI> element of the [RFC5730] <login> Command. The server includes the namespace URI in an <svcExtension> <extURI> element of the [RFC5730] Greeting.

A client that receives the namespace URI in the server's Greeting extension services can expect the following supported behavior by the server:

1. Support an empty authorization information value with a create command.
2. Support unsetting authorization information with an update command.
3. Support validating authorization information with an info command.
4. Support not returning an indication whether the authorization information is set or unset to the non-sponsoring registrar.

5. Support returning an empty authorization information value to the sponsoring registrar when the authorization information is set in an info response.
6. Support allowing for the passing of a matching non-empty authorization information value to authorize a transfer.
7. Support automatically unsetting the authorization information upon a successful completion of transfer.

A server that receives the namespace URI in the client's <login> Command extension services, can expect the following supported behavior by the client:

1. Support generation of authorization information using a secure random value.
2. Support only setting the authorization information when there is a transfer in process.

## 4. Secure Authorization Information

The authorization information in the EPP RFCs ([RFC5731] and [RFC5733]) that support transfer use password-based authorization information ([RFC5731] with the <domain:pw> element and [RFC5733] with the <contact:pw> element). 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 generated using a strong random value and have a short time-to-live (TTL). The security of the authorization information is defined in the following sections.

### 4.1. 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, and the required length L of a password, rounded up to the largest whole number, is based on the size N of the set of characters and the desired entropy H, in the equation L = ROUNDUP(H / log2 N). Given a target entropy, the required length can be calculated after deciding on the set of characters that will be randomized. In accordance with current best practices and noting that the authorization information is a machine-generated value, the implementation SHOULD use at least 128 bits of entropy as the value of H. The lengths below are calculated using that value.

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.

ROUNDUP(128 / log2 94) =~ ROUNDUP(128 / 6.55) =~ 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).

ROUNDUP(128 / log2 36) =~ ROUNDUP(128 / 5.17) =~ ROUNDUP(24.76) = 25

The strength of the random authorization information is dependent on the random number generator. Suitably strong random number generators are available in a wide variety of implementation environments, including the interfaces listed in Sections 7.1.2 and 7.1.3 of [RFC4086]. In environments that do not provide interfaces to strong random number generators, the practices defined in [RFC4086] and section 4.7.1 of the NIST Federal Information Processing Standards (FIPS) Publication 140-2 [FIPS-140-2] can be followed to produce random values that will be resistant to attack.

### 4.2. 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, 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.

### 4.3. Authorization Information Storage and Transport

To protect the disclosure of the authorization information, the following requirements apply:

1. The authorization information MUST be stored by the registry using a strong one-way cryptographic hash, with at least a 256-bit hash function, such as SHA-256 [FIPS-180-4], and with a per-authorization information random salt, with at least 128 bits.
2. Empty authorization information MUST be stored as an undefined value that is referred to as a NULL value. The representation of a NULL (undefined) value is dependent on the type of database used.
3. The authorization information MUST NOT be stored by the losing registrar.

4. The authorization information MUST only be stored by the gaining registrar as a "transient" value in support of the transfer process.
5. The plain text version of the authorization information MUST NOT be written to any logs by a registrar or the registry, nor otherwise recorded where it will persist beyond the transfer process.
6. All communication that includes the authorization information MUST be over an encrypted channel, such as defined in [RFC5734] for EPP.
7. The registrar's interface for communicating the authorization information with the registrant MUST be over an authenticated and encrypted channel.

### 4.4. Authorization Information Matching

To support the authorization information TTL, as defined in Section 4.2, the authorization information must have either a set or unset state. Authorization information that is unset is stored with a NULL (undefined) value. Based on the requirement to store the authorization information using a strong one-way cryptographic hash, as defined in Section 4.3, authorization information that is set is stored with a non-NULL hashed value. The empty authorization information is used as input in both the create command (Section 5.1) and the update command (Section 5.2) to define the unset state. The matching of the authorization information in the info command (Section 5.3) and the transfer request command (Section 5.4) is based on the following rules:

1. Any input authorization information value MUST NOT match an unset authorization information value. This includes empty authorization information, such as <domain:null/> or <domain:pw/> in [RFC5731], and non-empty authorization information, such as <domain:pw>2fooBAR</domain:pw> in [RFC5731].
2. An empty input authorization information value MUST NOT match any set authorization information value.
3. A non-empty input authorization information value MUST be hashed and matched against the set authorization information value, which is stored using the same hash algorithm.

## 5. Create, Transfer, and Secure Authorization Information

To secure the transfer process using secure authorization information, as defined in Section 4, 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 an empty authorization information value, to the registry, as defined in Section 5.1.
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 5.2, 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 5.3.
6. Gaining registrar sends the transfer request with the authorization information to the registry, as defined in Section 5.4.
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 5.2.

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.

### 5.1. Create Command

For a create command, the registry MUST allow for the passing of an empty authorization information value and MAY disallow for the passing of a non-empty authorization information value. By having an empty authorization information value 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 value passed. Examples of such extensions are [RFC5731] and [RFC5733].

Example of passing an empty authorization information value 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 an empty authorization information value 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:addr>
C:            <contact:city>Dulles</contact:city>
C:            <contact:cc>US</contact:cc>
C:          </contact:addr>
C:        </contact:postalInfo>
C:        <contact:email>jdoe@example.com</contact:email>
C:        <contact:authInfo>
C:          <contact:pw/>
C:        </contact:authInfo>
C:      </contact:create>
C:    </create>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

### 5.2. 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 4.1, and setting

it in the registry in the update command. The importance of generating strong authorization information values cannot be overstated: secure transfers are very important to the Internet to mitigate damage in the form of theft, fraud, and other abuse. It is critical that registrars only use strong, randomly generated authorization information values.

Because of this, registries may validate the randomness of the authorization information based on the length and character set required by the registry. For example, validating an authorization value contains a combination of upper-case, lower-case, and non-alphanumeric characters, in an attempt to assess the strength of the value, and return an EPP error result of 2202 if the check fails.

Such checks are, by their nature, heuristic and imperfect, and may identify well-chosen authorization information values as being not sufficiently strong. Registrars, therefore, must be prepared for an error response of 2202, "Invalid authorization information", and respond by generating a new value and trying again, possibly more than once.

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 4.2, 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 MUST cancel the transfer process by unsetting the authorization information value and MAY add back statuses like the "clientTransferProbited" status. Any EPP object extension that supports setting the authorization information with a "eppcom:pwAuthInfoType" element, can have an empty authorization information value passed. Examples of such extensions are [RFC5731] and [RFC5733]. Setting an empty authorization information value unsets the authorization information. [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 adding the "clientTransferProhibited" status and 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 value 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 value 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>
```

### 5.3. Info Command and Response

For an info command, the registry MUST allow for the passing of a non-empty authorization information value 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 value 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 with a non-empty authorization value. The authorization information is stored as a hash in the registry, so returning the plain text authorization information is not possible, unless a valid plain text authorization information is passed in the info command. The registry MUST NOT return any indication of whether the authorization information is set or unset to the non-sponsoring registrar by not returning the authorization information element in the response. The registry MAY return an indication to the sponsoring registrar that the authorization information is set by using an empty authorization information value. The registry MAY return an indication to the sponsoring registrar that the authorization information is unset by not returning the authorization information element.

Example of returning an empty authorization information value in an [RFC5731] domain name info response to indicate to the sponsoring registrar that the authorization information is set:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <domain:infData
S:       xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:        <domain:name>example.com</domain:name>
S:        <domain:roid>EXAMPLE1-REP</domain:roid>
S:        <domain:status s="ok"/>
S:        <domain:clID>ClientX</domain:clID>
S:        <domain:authInfo>
S:          <domain:pw/>
S:        </domain:authInfo>
S:      </domain:infData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

### 5.4. Transfer Request Command

For a Transfer Request Command, the registry MUST allow for the passing of a non-empty authorization information value 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 value 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 request is not submitted within the time-to-live (TTL) (Section 4.2) or the transfer is cancelled or rejected, the registrar MUST unset the authorization information as defined in Section 5.2.

## 6. Transition Considerations

The goal of the transition considerations to the practice defined in this document, referred to as the Secure Authorization Information Model, is to minimize the impact to the registrars by supporting incremental steps of adoption. The transition steps are dependent on the starting point of the registry. Registries may have different starting points, since some of the elements of the Secure Authorization Information Model may have already been implemented. The considerations assume a starting point, referred to as the Classic Authorization Information Model, that have the following steps in the management of the authorization information for transfers:

1. Registrant requests to register the object with the registrar. Registrar sends the create command, with a non-empty authorization information value, to the registry. The registry stores the authorization information as an encrypted value and requires a non-empty authorization information value for the life of the object. The registrar may store the long-lived authorization information.
2. At the time of transfer, Registrant requests from the losing registrar the authorization information to provide to the gaining registrar.
3. Losing registrar retrieves the locally stored authorization information or queries the registry for authorization

information using the info command, and provides it to the registrant. If the registry is queried, the authorization information is decrypted and the plain text authorization information is returned in the info response to the registrar.
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, by passing the authorization information in the info command to the registry.
6. Gaining registrar sends the transfer request with the authorization information to the registry. The registry will decrypt the stored authorization information to compare to the passed authorization information.
7. If the transfer successfully completes, the authorization information is not touched by the registry and may be updated by the gaining registrar using the update command. If the transfer is cancelled or rejected, the losing registrar may reset the authorization information using the update command.

The gaps between the Classic Authorization Information Model and the Secure Authorization Information Model include:

1. Registry requirement for a non-empty authorization information value on create and for the life of the object versus the authorization information not being set on create and only being set when a transfer is in process.
2. Registry not allowing the authorization information to be unset versus supporting the authorization to be unset in the update command.
3. Registry storing the authorization information as an encrypted value versus as a hashed value.
4. Registry support for returning the authorization information versus not returning the authorization information in the info response.
5. Registry not touching the authorization information versus the registry automatically unsetting the authorization information upon a successful transfer.
6. Registry may validate a shorter authorization information value using password complexity rules versus validating the randomness of a longer authorization information value that meets the required bits of entropy.

The transition can be handled in the three phases defined in the sub-sections Section 6.1, Section 6.2, Section 6.3.

### 6.1. Transition Phase 1 - Features

The goal of the "Transition Phase 1 - Features" is to implement the needed features in EPP so that the registrar can optionally implement the Secure Authorization Information Model. The features to implement are broken out by the command and responses below:

**Create Command:** Change the create command to make the authorization information optional, by allowing both a non-empty value and an empty value. This enables a registrar to optionally create objects without an authorization information value, as defined in Section 5.1.

**Update Command:** Change the update command to allow unsetting the authorization information, as defined in Section 5.2. This enables the registrar to optionally unset the authorization information when the TTL expires or when the transfer is cancelled or rejected.

**Transfer Approve Command and Transfer Auto-Approve:** Change the transfer approve command and the transfer auto-approve to automatically unset the authorization information. This sets the default state of the object to not have the authorization information set. The registrar implementing the Secure Authorization Information Model will not set the authorization information for an inbound transfer and the registrar implementing the Classic Authorization Information Model will set the new authorization information upon the successful transfer.

**Info Response:** Change the info command to not return the authorization information in the info response, as defined in Section 5.3. This sets up the implementation of "Transition Phase 2 - Storage", since the dependency in returning the authorization information in the info response will be removed. This feature is the only one that is not an optional change to the registrar that has the potential of breaking the client, so it's recommended that the registry provide notice of the change.

**Info Command and Transfer Request:** Change the info command and the transfer request to ensure that a registrar cannot get an indication that the authorization information is set or not set by returning the EPP error result code of 2202 when comparing a passed authorization to a non-matching set authorization information value or an unset value.

### 6.2. Transition Phase 2 - Storage

The goal of the "Transition Phase 2 - Storage" is to transition the registry to use hashed authorization information instead of encrypted authorization information. There is no direct impact to the registrars, since the only visible indication that the authorization information has been hashed is by not returning the set authorization information in the info response, which is

addressed in Transition Phase 1 - Features (Section 6.1). There are three steps to transition the authorization information storage, which includes:

**Hash New Authorization Information Values:** Change the create command and the update command to hash instead of encrypting the authorization information.

**Supporting Comparing Against Encrypted and Hashed Authorization Information:**
Change the info command and the transfer request command to be able to compare a passed authorization information value with either a hashed or encrypted authorization information value. This requires that the stored values are self-identifying as being in hashed or encrypted form.

**Hash Existing Encrypted Authorization Information Values:** Convert the encrypted authorization information values stored in the registry database to hashed values. The update is not a visible change to the registrar. The conversion can be done over a period of time depending on registry policy.

### 6.3. Transition Phase 3 - Enforcement

The goal of the "Transition Phase 3 - Enforcement" is to complete the implementation of the "Secure Authorization Information Model", by enforcing the following:

**Disallow Authorization Information on Create Command:** Change the create command to not allow for the passing of a non-empty authorization information value. This behavior has the potential of breaking the client, so it's recommended that the registry provide notice of the change.

**Validate the Strong Random Authorization Information:** Change the validation of the authorization information in the update command to ensure at least 128 bits of entropy.

## 7. IANA Considerations

### 7.1. XML Namespace

This document uses URNs to describe XML namespaces conforming to a registry mechanism described in [RFC3688]. The following URI assignment is requested of IANA:

Registration request for the secure authorization information for transfer namespace:

URI: urn:ietf:params:xml:ns:epp:secure-authinfo-transfer-1.0
Registrant Contact: IESG
XML: None. Namespace URIs do not represent an XML specification.

### 7.2. EPP Extension Registry

The EPP operational practice described in this document should be registered by the IANA in the EPP Extension Registry described in [RFC7451]. The details of the registration are as follows:

Name of Extension: "Extensible Provisioning Protocol (EPP) Secure Authorization Information for Transfer"

Document status: Standards Track

Reference: (insert reference to RFC version of this document)

Registrant Name and Email Address: IESG, <iesg@ietf.org>

TLDs: Any

IPR Disclosure: None

Status: Active

Notes: None

## 8. 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".

### 8.1. 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-ietf-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

URL: https://www.verisign.com/en_US/channel-resources/domain-registry-products/epp-sdks

### 8.2. 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: Authorization Information is "write only" in that the registrars can set the Authorization Information, but not get the Authorization Information in the Info Response.

Licensing: Proprietary In-House software

Contact: epp@centralnic.com

URL: https://www.centralnic.com

## 9. Security Considerations

Section 4.1 defines the use a secure random value for the generation of the authorization information. The client SHOULD choose a length and set of characters that results in at least 128 bits of entropy.

Section 4.2 defines the use of an authorization information Time-To-Live (TTL). The registrar SHOULD only set the authorization information during the transfer process by the server support for setting and unsetting the authorization information. The TTL value is up to registrar policy and the sponsoring registrar MUST inform the registrant of the TTL when providing the authorization information to the registrant.

Section 4.3 defines the storage and transport of authorization information. The losing registrar MUST NOT store the authorization information and the gaining registrar MUST only store the authorization information as a "transient" value during the transfer process, where the authorization information MUST NOT be stored after the end of the transfer process. The registry MUST store the authorization information using a one-way cryptographic hash of at least 256 bits and with a per-authorization information random salt, with at least 128 bits. All communication that includes the authorization information MUST be over an encrypted channel. The plain text authorization information MUST NOT be written to any logs by the registrar or the registry.

Section 4.4 defines the matching of the authorization information values. The registry stores an unset authorization information as a NULL (undefined) value to ensure that an empty input authorization information never matches it. The method used to define a NULL (undefined) value is database specific.

## 10. Acknowledgements

The authors wish to thank the following persons for their feedback and suggestions: Michael Bauland, Martin Casanova, Scott Hollenbeck, Benjamin Kaduk, Jody Kolker, Barry Leiba, Patrick Mevzek, Matthew Pozun, Srikanth Veeramachaneni, and Ulrich Wisser.

## 11. References

### 11.1. 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>.

**[RFC3688]** Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

**[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>.

**[RFC8174]** Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

**[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>.

### 11.2. Informative References

**[FIPS-140-2]** National Institute of Standards and Technology, U.S. Department of Commerce, "NIST Federal Information Processing Standards (FIPS) Publication 140-2", May 2001, <https://csrc.nist.gov/publications/detail/fips/140/2/final>.

**[FIPS-180-4]** National Institute of Standards and Technology, U.S. Department of Commerce, "Secure Hash Standard, NIST Federal Information Processing Standards (FIPS)

Publication 180-4", August 2015, <https://csrc.nist.gov/publications/detail/fips/180/4/final>.

[RFC7451] Hollenbeck, S., "Extension Registry for the Extensible Provisioning Protocol", RFC 7451, DOI 10.17487/RFC7451, February 2015, <https://www.rfc-editor.org/info/rfc7451>.

## 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.

### A.3. Change from 02 to 03

1. Updates based on the feedback from the interim REGEXT meeting held at ICANN-66:
    1. Section 3.3, include a reference to the hash algorithm to use. Broke the requirements into a list and included a the reference the text ', with at least a 256-bit hash function, such as SHA-256'.
    2. Add a Transition Considerations section to cover the transition from the classic authorization information security model in the EPP RFCs to the model defined in the document.
    3. Add a statement to the Introduction that elements of the practice can be used for purposes other than transfer, but with a caveat.
2. Updates based on the review by Michael Bauland, that include:
    1. In section 2, change 'there are three actors' to 'there are three types of actors' to cover the case with transfers that has two registrar actors (losing and gaining).
    2. In section 3.1, change the equations equals to be approximately equal by using '=~' instead of '=', where applicable.

3. In section 3.3, change 'MUST be over an encrypted channel, such as RFC5734' to 'MUST be over an encrypted channel, such as defined in RFC5734'.
4. In section 4.1, remove the optional RFC 5733 elements from the contact create, which includes the <contact:voice>, <contact:fax>, <contact:disclose>, <contact:org>, <contact:street>, <contact:sp>, and <contact:cc> elements.
5. In section 4.2, changed 'Example of unsetting the authorization information explicitly in an [RFC5731] domain name update command.' to 'Example of adding the "clientTransferProhibited" status and unsetting the authorization information explicitly in an [RFC5731] domain name update command.'
6. In section 4.3, cover a corner case of the ability to return the authorization information when it's passed in the info command.
7. In section 4.4, change 'If the transfer does not complete within the time-to-live (TTL)' to 'If the transfer is not initiated within the time-to-live (TTL)', since the TTL is the time between setting the authorization information and when it's successfully used in a transfer request. Added the case of unsetting the authorization information when the transfer is cancelled or rejected.

3. Updates based on the authorization information messages by Martin Casanova on the REGEXT mailing list, that include:

    1. Added section 3.4 'Authorization Information Matching' to clarify how the authorization information is matched, when there is set and unset authorization information in the database and empty and non-empty authorization information passed in the info and transfer commands.
    2. Added support for signaling that the authorization information is set or unset to the sponsoring registrar with the inclusion of an empty authorization information element in the response to indicate that the authorization information is set and the exclusion of the authorization information element in the response to indicate that the authorization information is unset.

4. Made the capitalization of command and response references consistent by uppercasing section and item titles and lowercasing references elsewhere.

### A.4. Change from 03 to REGEXT 00

1. Changed to regext working group draft by changing draft-gould-regext-secure-authinfo-transfer to draft-ietf-regext-secure-authinfo-transfer.

### A.5. Change from REGEXT 00 to REGEXT 01

1. Added the "Signaling Client and Server Support" section to describe the mechanism to signal support for the BCP by the client and the server.
2. Added the "IANA Considerations" section with the registration of the secure authorization for transfer XML namespace and the registration of the EPP Best Current Practice (BCP) in the EPP Extension Registry.

### A.6. Change from REGEXT 01 to REGEXT 02

1. Added inclusion of random salt for the hashed authorization information, based on feedback from Ulrich Wisser.
2. Added clarification that the representation of a NULL (undefined) value is dependent on the type of database, based on feedback from Patrick Mevzek.
3. Filled in the Security Considerations section.

### A.7. Change from REGEXT 02 to REGEXT 03

1. Updated the XML namespace to urn:ietf:params:xml:ns:epp:secure-authinfo-transfer-1.0, which removed bcp from the namespace and bumped the version from 0.1 and 1.0. Inclusion of bcp in the XML namespace was discussed at the REGEXT interim meeting.
2. Replaced Auhtorization with Authorization based on a review by Jody Kolker.

### A.8. Change from REGEXT 03 to REGEXT 04

1. Converted from xml2rfc v2 to v3.
2. Updated Acknowledgements to match the approach taken by the RFC Editor with draft-ietf-regext-login-security.
3. Changed from Best Current Practice (BCP) to Standards Track based on mailing list discussion.

### A.9. Change from REGEXT 04 to REGEXT 05

1. Fixed IDNITS issues, including moving RFC7451 to Informative References section.

### A.10. Change from REGEXT 05 to REGEXT 06

Updates based on the Barry Leiba (AD) feedback:

1. Simplified the abstract based on the proposal provided.
2. In the Introduction, split the first paragraph by starting a new paragraph at "This document".
3. In section 1.1, updated to use the new BCP 14 boilerplate and add a normative reference to RFC 8174.

4. In section 4, Updated the phrasing to "For the authorization information to be secure it must be generated using a strong random value and have a short time-to-live (TTL).".
5. In section 4.1, removed the first two unnecessary calculations and condensed the introduction of the section.
6. In section 4.1, added the use of the normative SHOULD for use of at least 128 bits of entropy.
7. Added an informative reference to FIPS 180-4 for the SHA-256 references.
8. Normalized the way that the "empty and non-empty authorization information values" are referenced, which a few exceptions.
9. In section 4, revised the first sentence to explicitly reference the use of the <domain:pw> and <contact:pw> elements for password-based authorization information.
10. In section 4.4, revised the language associated with the storage of the authorization information to be cleaner.
11. In section 4.4, added "set" in the sentence "An empty input authorization information value MUST NOT match any set authorization information value."
12. In section 5.1 and 5.2, clarified the references to RFC5731 and RFC5733 as examples of object extensions that use the "eppcom:pwAuthInfoType" element.
13. In section 5.2, updated language for the validation of the randomness of the authorization information, based on an offline review by Barry Leiba, Benjamin Kaduk, and Roman Danyliw.
14. In section 9, changed "49 bits of entropy" to "128 bits of entropy".

In section 3, replaced the reference to BCP with operational practice, since the draft is not defined as a BCP.

### A.11. Change from REGEXT 06 to REGEXT 07

1. Updates based on the Lars Eggert feedback:

    1. Updated Section 1, Paragraph 4 to read "The operational practice will require the client to not store the authorization information and".
    2. Updated each of the example references to end with a colon instead of a period.
    3. Updated Section 1, Paragraph 3 to read "provide secure authorization information used for transfers."
    4. Updated Section 3, Paragraph 3 to read "extension services can expect".
    5. Updated Section 4, Paragraph 2 to read "authorization information to be secure, it must".
    6. Updated Section 4.2, Paragraph 2 to read "authorization information by the sponsoring registrar, the".

7. Updated Section 4.2, Paragraph 2 to read "proprietary registrar-specific criteria, which".
8. Updated Section 4.3, Paragraph 3 to read "256-bit hash function, such as SHA-256".
9. Updated Section 4.3, Paragraph 3 to read "a NULL (undefined) value".
10. Updated Section 5, Paragraph 2 to read "To secure the transfer process using secure authorization".
11. Updated Section 5.2, Paragraph 6 to read "Often, the registrar has the "clientTransferProhibited" status set".
12. Updated Section 5.2, Paragraph 9 to read "MUST cancel cancel the transfer process by unsetting the authorization information value and MAY add back statuses".
13. Updated Section 5.2, Paragraph 9 to read ""eppcom:pwAuthInfoType" element can have".

2. Updated the first sentence of the abstract and introduction based on the Rob Wilton feedback to help non-EPP readers on the what and the who for transfers.

3. Removed the duplicate first paragraph of section 5.2 based on feedback from Francesca Palombini.

4. Updates based on the Benjamin Kaduk feedback:

    1. Added the second paragraph in the Introduction to provide high-level motivation for the work.
    2. Updated Section 1, changed "in any way" to "in any substantial way".
    3. Updated Section 1 by adding the sentence "All of these features are compatible with the EPP RFCs, though not mandatory to implement." for the "Short-Lived Authorization Information".
    4. Updated the description of "Short-Lived Authorization Information" in Section 1 to reference section 2.6 of RFC5731 and change in nature of the authorization information.
    5. Updated Section 4.1, Paragraph 1 and 2 were merged with modified language proposed by Benjamin Kaduk, which included removing the reference to RFC4086 for length and entropy.
    6. Updated rule #1 of Section 4.1 to add a second clarifying sentence for what is meant by input authorization information.
    7. Updated Section 4.1 by replacing the last paragraph "The strength of the random..." with a revised version.
    8. Updated "retrieves the stored authorization information locally" with "retrieves the locally stored authorization information".

9. Updated Section 6.1 to include the recommendation that the registry provide notice of the Info Response change.
10. Updated Section 6.2 to include the sentence "This requires that the stored values are self-identifying as being in hashed or encrypted form" for the "Supporting Comparing Against Encrypted and Hashed Authorization Information" step.
11. Updated Section 6.3 to include the recommendation that the registry provide notice of the Create Command change.
12. Updated "written to any logs by the registrar or the registry" to "written to any logs by a registrar or the registry" to cover both the losing and the gaining registrar.
13. Updated references to "with a random salt" to "with a per-authorization information random salt, with at least 128 bits" to address sharing of salts and the size of the salts.
14. Updated the first paragraph of Section 9 to remove the reference to defining a server policy for the length and set of characters that are included in the randomization to target the target entropy level.
15. Updated Section 9 by removing the sentence "A random number generator (RNG) is preferable over the use of a pseudorandom number generator (PRNG) when creating the authorization information value."
16. Changed FIPS-140-2 from a normative reference to an informative reference.

## Authors' Addresses


James Gould
VeriSign, Inc.
12061 Bluemont Way
Reston, VA 20190
United States of America

Email: jgould@verisign.com
URI: http://www.verisign.com

Richard Wilhelm
VeriSign, Inc.
12061 Bluemont Way
Reston, VA 20190
United States of America

Email: rwilhelm@verisign.com
URI: http://www.verisign.com