

# Extensible Provisioning Protocol (EPP) v1.6.1 .INFO Registrar Acceptance Criteria



Published May 18, 2011  
Version 1.6.1

Technical Support:  
Portal: [https://registrars.afilias.info/cgi-bin/ssl/open\\_update\\_ticket.cgi](https://registrars.afilias.info/cgi-bin/ssl/open_update_ticket.cgi)  
Phone: +1.416.646.3306  
URL: <http://www.afilias.info>

# Contents

## 1. Introduction

- 1.1 Purpose
- 1.2 Formatting Conventions
- 1.3 Accounts
- 1.4 Additional Requirements
- 1.5 Successful Command & Test Completion
- 1.6 Passing the Test
- 1.7 Contact and Name Server Policy Requirements

## 2. EPP Communications

### 2.1 Starting the Test

### 2.2 Session Management

- 2.2.1 Start Session
- 2.2.2 Authentication
- 2.2.3 Change Password

### 2.3 Mandatory EPP Acceptance Criteria

#### 2.3.1 Creation of Objects and their Updates

- 2.3.1.1 Check Contact OTE-C1 (Contact Available)
- 2.3.1.2 Create Contact OTE-C1
- 2.3.1.3 Check Contact OTE-C1 (Contact Not Available)
- 2.3.1.4 Query Contact OTE-C1
- 2.3.1.5 Check Contact OTE-C2 (Contact Available)
- 2.3.1.6 Create Contact OTE-C2
- 2.3.1.7 Check Contact OTE-C3 (Contact Available)
- 2.3.1.8 Create Contact OTE-C3
- 2.3.1.9 Check Contact OTE-C4 (Contact Available)
- 2.3.1.10 Create Contact OTE-C4
- 2.3.1.11 Update Contact (Change Element)
- 2.3.1.12 Update Contact (Remove Element)
- 2.3.1.13 Update Contact (Add Element)
- 2.3.1.14 Check Name Server (Foreign Registry - Available)
- 2.3.1.15 Create Name Server (Foreign Registry)
- 2.3.1.16 Check Name Server (Foreign Registry - Available)
- 2.3.1.17 Create Name Server (Foreign Registry)
- 2.3.1.18 Check Domain (Domain Available for Registration)
- 2.3.1.19 Create Domain (example.info)
- 2.3.1.20 Check Domain (Domain Not Available for Registration)
- 2.3.1.21 Query Domain
- 2.3.1.22 Check Name Server (Available)
- 2.3.1.23 Create Name Server

- 2.3.1.24 Check Name Server (Unavailable)
- 2.3.1.25 Query Name Server
- 2.3.1.26 Check Name Server (Available)
- 2.3.1.27 Create Name Server
- 2.3.1.28 Update Name Server (Add IP Address)
- 2.3.1.29 Update Name Server (Remove IP Address)
- 2.3.1.30 Check Domain (Domain Available for Registration)
- 2.3.1.31 Create Domain (domain.info)
- 2.3.1.32 Query Domain
- 2.3.1.33 Renew Domain
- 2.3.1.34 Update Domain – Change Name Servers
- 2.3.1.35 Update Domain - Change Contact
- 2.3.1.36 Update Domain – Change Authorization Information
- 2.3.1.37 Update Domain - Change Domain Status

## **2.3.2 Transfer of Objects**

- 2.3.2.1 Contact Transfer Request
- 2.3.2.2 Query Contact Transfer
- 2.3.2.3 Approve Contact Transfer
- 2.3.2.4 Reject Contact Transfer
- 2.3.2.5 Domain Transfer Request
- 2.3.2.6 Approve Domain Transfer
- 2.3.2.7 Reject Domain Transfer

## **2.3.3 Client Error Handling**

- 2.3.3.1 Correctly Handle 2003 Exception
- 2.3.3.2 Correctly Handle 2005 Exception
- 2.3.3.3 Correctly Handle 2306 Exception
- 2.3.3.4 Correctly Handle 2002 Exception
- 2.3.3.5 Correctly Handle 2303 Exception
- 2.3.3.6 Correctly Handle 2005 Exception
- 2.3.3.7 Correctly Handle 2201 Exception

## **2.4 IDN EPP Acceptance Criteria (Optional)**

### **2.4.1 Creation of IDN objects and their updates**

- 2.4.1.1 Check Domain (Domain Available for Registration)
- 2.4.1.2 Create IDN Domain using Hungarian script
- 2.4.1.3 Check Domain (Domain Not Available for Registration)
- 2.4.1.4 Query Hungarian IDN Domain
- 2.4.1.5 Update Domain with Supported IDN Script
- 2.4.1.6 Check Multiple IDN with proper Script
- 2.4.1.7 Check IDN with invalid script

### **2.4.2 Client Error Handling**

- 2.4.2.1 Correctly Handle 2003 Error Exception
- 2.4.2.2 Correctly Handle 2003 Error Exception
- 2.4.2.3 Correctly Handle 2306 Error Exception
- 2.4.2.4 Correctly Handle 2306 Error Exception

### **2.4.3 Delete IDN Objects**

- 2.4.3.1 Delete IDN Domain
- 2.4.3.2 Delete Contact (OTE-C1)
- 2.4.3.3 Delete Contact (OTE-C2)
- 2.4.3.4 Delete Contact (OTE-C3)
- 2.4.3.5 Delete Contact (OTE-C4)
- 2.4.3.6 Delete Name Server (ns1.example.com)
- 2.4.3.7 Delete Name Server (ns2.example.com)

## **2.5 Deletion of Other Objects**

- 2.5.1 Delete Domain (example.info)
- 2.5.2 Delete Domain (domain.info)
- 2.5.3 Delete Contact (OTE-C1)
- 2.5.4 Delete Contact (OTE-C2)
- 2.5.5 Delete Contact (OTE-C3)
- 2.5.6 Delete Contact (OTE-C4)
- 2.5.7 Delete Name Server (ns1.example.com)
- 2.5.8 Delete Name Server (ns2.example.com)

## **2.6 Efficiency of Client Session Management**

- 2.6.1 Keep Session Alive
- 2.6.2 Request Message Queue Information
- 2.6.3 Ack Queued Message

## **2.7 End Session**

## **2.8 Completing the Test**

# **Appendix A -Seeded Registry information**

*This document is made available to the registrars that have entered into Registry-Registrar Agreements with Afilias, manager of the registry of .INFO. The contents of this document are proprietary information of Afilias, Limited. This information may be used by recipient only for the purpose for which it was transmitted and shall be returned upon request to Afilias Limited or when no longer needed by recipient.*

*Afilias has made efforts to ensure the accuracy and completeness of the information in this document. However, Afilias makes no warranties of any kind (express or implied) with respect to the information contained herein. Afilias assumes no liability to any party for any loss or damage (whether direct or indirect) caused by any errors, omissions, or statements of any kind contained in this document. Afilias reserves the right to make changes to any information herein at any time, without further notice.*

# 1. Introduction

## 1.1 Purpose

This document describes the basic operations that a Registrar's client application must perform to be accepted by the Registry. Each of the following sections describes the actions that the client must perform to demonstrate correct implementation of the Extensible Provisioning Protocol (EPP) v1.0 and interactions with the Registry. Registrars should have a detailed knowledge of the following internet RFCs before attempting the test:

*EPP RFC: 5730* (<http://www.rfc-editor.info/rfc/rfc5730.txt>)

*EPP Domain Name Mapping RFC: 5731* (<http://www.rfc-editor.info/rfc/rfc5731.txt>)

*EPP Host Mapping RFC: 5732* (<http://www.rfc-editor.info/rfc/rfc5732.txt>)

*EPP Contact Mapping RFC: 5733* (<http://www.rfc-editor.info/rfc/rfc5733.txt>)

*EPP Transport Over TCP RFC: 5734* (<http://www.rfc-editor.info/rfc/rfc5734.txt>)

The tests presented herein verify the correct interface with the Registry for standard Registrar operations. They do not cover all possible error and unusual conditions. The Registrar client application is responsible for correctly handling all unusual error conditions.

## 1.2 Formatting Conventions

Proper completion of the test requires that all commands and data must be entered exactly as given in this document. Any deviations will be considered a failure. The following items show the formatting conventions included in this document for required input and output values and for variable input and output responses.

Regular text in this format represents expected system input and output values that the client system will send to the server and that the server system will display in response to an action or actions provided by the Registrar. The following example illustrates an expected system output.

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

When **bold** text is located in Regular text, this represents a required input value that the Registrar must provide -the Registrar must enter the text exactly as shown. The following example illustrates the format for the required input values.

Domain Name: **example.info**

Italicized text in output represents data returned from the server, which may or may not be the exact values represented in this document. It is the responsibility of the client to interpret these values properly and possibly reuse these for subsequent commands.

```
<domain:exDate>2011-06-21T22:07:28.0Z</domain:exDate>
```

## 1.3 Accounts

For the duration of the test, the Registrar will use a seeded test account, called ClientX. The Registrar

will provide AFILIAS Technical Support Group with a valid email address. Standard registry transfer notifications, processed by the registry during the initial test seeding (\*\*see appendix for details\*\*), will be sent to this e-mail address for Registrar reference. Upon the scheduling of a test, Afiliias Technical Support Group will provide hostnames and port numbers for the Registrar's client connection.

## **1.4 Additional Requirements**

Registry Operator will prime the Test Registry with data required to complete this test. Please refer to Appendix A if you wish to review this data. Do not attempt to enter this data into the Test Registry.

## **1.5 Successful Command & Test Completion**

While performing this test, if the response to a command is not exactly as shown, then stop your test and contact AFILIAS Technical support.

## **1.6 Passing the Test**

The Registrar must complete the test perfectly (with no typographical errors and without breaking the sequence of operations) from start to finish within the allotted time.

## **1.7 Contact and Name Server Policy Requirements**

There are certain policies that are enforced in the .INFO implementation of EPP:

A minimum of 4 contacts (including 1 Registrant and at least 1 of each Admin, Billing and Technical contacts) must be provided during the create domain transaction.

For the purpose of this test, all domains must be created with at least 2 name servers. Registrars may, however, when working with the "live" registry, create domains with fewer than 2 name servers, though DNS resolution depends upon a minimum of one (1) assigned name server. The use of at least two (2) valid nameservers is highly recommended.

## **2. EPP Communications**

Registrar to Registry communications utilize the Extensible Provisioning Protocol (EPP) mapped over TCP (Transport Control Protocol). EPP commands are formulated using the Extensible Markup Language (XML). The Registrars' application client must utilize XML to send commands to the Registry and utilize an XML parser to interpret the server's responses. EPP itself relies exclusively upon user authentication for security. Additional security is provided by the use of Transport Layer Security (TLS), for session cryptography. Clients must communicate with the EPP server using a commercial or open source implementation of TLS, such as OpenSSL. Additional information concerning mapping EPP over TCP is available in 'RFC 5734 - Extensible Provisioning Protocol Transport Over TCP'. Additional information concerning the TLS may be found in RFC 5246.

## 2.1 Starting the Test

AFILIAS Technical Support will contact the Registrar by telephone a few minutes before the scheduled start time, to provide final confirmation prior to the Registrar commencing the OT&E test.

## 2.2 Session Management

### 2.2.1 Start Session

After making an initial connection to the Registry, the server shall reply with a greeting. A Registrar must receive the greeting message before attempting authentication and/or other supplementary commands.

### 2.2.2 Authentication

After the initial greeting the Registrar client shall send the Login command to authenticate itself to the test registry with the following information:

Client ID: **ClientX**  
Password: **foo-BAR2**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.2.3 Change Password

To change a Registrar's password, an additional field is required in the Login command. At this point, the client must log out, then log in again, and pass the following information to the Login command:

Client ID: **ClientX**  
Password: **foo-BAR2**  
New Password: **bar-FOO2**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

## 2.3 Mandatory EPP Acceptance Criteria

To pass the regular EPP OT&E test, a Registrar has to perform all tests listed under sections, **2.3.1**, **2.3.2**, **2.3.3**, **2.6**, **2.7**, **2.8** and **2.9**.

### 2.3.1 Creation of Objects and their Updates

The following tests exercise EPP commands that revolve around object creation and updates.

### 2.3.1.1 Check Contact OTE-C1 (Contact Available)

Use the Check command with the following argument.

ID: **OTE-C1**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg><contact:id avail='1'>
```

### 2.3.1.2 Create Contact OTE-C1

Supply the following information to the Create command.

Contact ID: **OTE-C1**  
Contact Name: **John Doe**  
Contact Organization: **Example Corp. Inc**  
Contact Address Street1: **123 Example St.**  
Contact Address Street2: **Suite 100**  
Contact Address City: **Anytown**  
Contact Address State/Province: **Any Prov**  
Contact Address Postal Code: **A1A1A1**  
Contact Address Country: **CA**  
Contact Voice: **+1.4165555555**  
Contact Voice Extension: **1111**  
Contact Fax: **+1.4165555556**  
Contact Email: **jdoe@test.test**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.1.3 Check Contact (Contact Not Available)

Use the Check command with the following argument.

ID: **OTE-C1**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg><contact:id avail='0'>
```

### 2.3.1.4 Query Contact OTE-C1

Supply the following information to the Info command.

ID: **OTE-C1**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

Contact ID: **OTE-C1**  
Contact Name: **John Doe**  
Contact Organization: **Example Corp. Inc**  
Contact Address Street1: **123 Example St.**  
Contact Address Street2: **Suite 100**  
Contact Address City: **AnyTown**  
Contact Address State/Province: **AnyProvince**  
Contact Address Postal Code: **A1A1A1**  
Contact Address Country: **CA**  
Contact Voice: **+1.4165555555**  
Contact Voice Extension: **1111**  
Contact Fax: **+1.4165555556**  
Contact Email: [jdoe@test.test](mailto:jdoe@test.test)  
Auth Info: **my secret**  
Status: **ok**

**Note: Create 3 more contacts, OTE-C2 to OTE-C4, to be used for domain operations.**

#### **2.3.1.5 Check Contact OTE-C2 (Contact Available)**

Use the Check command with the following argument.

ID: **OTE-C2**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg><contact:id  
avail='1'>
```

#### **2.3.1.6 Create Contact OTE-C2**

Supply the following information to the Create command.

Contact ID: **OTE-C2**  
Contact Name: **John Doe**  
Contact Organization: **Example Corp. Inc**  
Contact Address Street1: **123 Example St.**  
Contact Address Street2: **Suite 100**  
Contact Address City: **Anytown**  
Contact Address State/Province: **Any Prov**  
Contact Address Postal Code: **A1A1A1**  
Contact Address Country: **CA**  
Contact Voice: **+1.4165555555**

Contact Voice Extension: **1111**  
Contact Fax: **+1.4165555556**  
Contact Email: **jd@test.test**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### **2.3.1.7 Check Contact OTE-C3 (Contact Available)**

Use the Check command with the following argument.

ID: **OTE-C3**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg><contact:id  
avail='1'>
```

### **2.3.1.8 Create Contact OTE-C3**

Supply the following information to the Create command.

Contact ID: **OTE-C3**  
Contact Name: **John Doe**  
Contact Organization: **Example Corp. Inc**  
Contact Address Street1: **123 Example St.**  
Contact Address Street2: **Suite 100**  
Contact Address City: **Anytown**  
Contact Address State/Province: **Any Prov**  
Contact Address Postal Code: **A1A1A1**  
Contact Address Country: **CA**  
Contact Voice: **+1.4165555555**  
Contact Voice Extension: **1111**  
Contact Fax: **+1.4165555556**  
Contact Email: **jd@test.test**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### **2.3.1.9 Check Contact OTE-C4 (Contact Available)**

Use the Check command with the following argument.

ID: **OTE-C4**

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg><contact:id avail='1'>

#### 2.3.1.10 Create Contact OTE-C4

Supply the following information to the Create command.

Contact ID: **OTE-C4**  
Contact Name: **John Doe**  
Contact Organization: **Example Corp. Inc**  
Contact Address Street1: **123 Example St.**  
Contact Address Street2: **Suite 100**  
Contact Address City: **Anytown**  
Contact Address State/Province: **Any Prov**  
Contact Address Postal Code: **A1A1A1**  
Contact Address Country: **CA**  
Contact Voice: **+1.4165555555**  
Contact Voice Extension: **1111**  
Contact Fax: **+1.4165555556**  
Contact Email: **jdoe@test.test**  
Auth Info: **my secret**

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg>

#### 2.3.1.11 Update Contact (Change Element)

Supply the following information to the Update command.

ID: **OTE-C3**  
Contact Name: **Jane Smith**

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg>

#### 2.3.1.12 Update Contact (Remove Element)

Supply the following information to the Update command. To remove a value, overwrite it as a NULL value.

ID: **OTE-C3**  
Contact Fax:

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg>

### 2.3.1.13 Update Contact (Add Element)

Supply the following information to the Update command.

ID: **OTE-C3**  
Contact Fax: **+1.4165555556**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.1.14 Check Name Server (Foreign Registry - Available)

Supply the following to the Check command.

ID: Host Name: **ns1.example.com**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg><contact:id  
avail='1'>
```

### 2.3.1.15 Create Name Server (Foreign Registry)

Supply the following to the Create command:

Host Name: **ns1.example.com**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.1.16 Check Name Server (Foreign Registry - Available)

Supply the following to the Check command.

ID: Host Name: **ns2.example.com**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg><contact:id  
avail='1'>
```

### 2.3.1.17 Create Name Server (Foreign Registry)

Supply the following to the Create command:

Host Name: **ns2.example.com**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### **2.3.1.18 Check Domain (Domain Available for Registration)**

Use the Check command with the following data to determine that the domain is available:

Domain Name: **example.info**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>  
domain:name avail='1'
```

### **2.3.1.19 Create Domain**

Create a new domain and associate two (2) Name Servers and four (4) Contacts to it by supplying the following elements to the Create command.

Domain Name: **example.info**  
Domain Server: **ns1.example.com**  
Domain Server: **ns2.example.com**  
Domain Registrant Contact ID: **OTE-C1**  
Domain Admin Contact ID: **OTE-C2**  
Domain Billing Contact ID: **OTE-C3**  
Domain Technical Contact ID: **OTE-C4**  
Domain Registration Period (Year): **1**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### **2.3.1.20 Check Domain (Domain Not Available for Registration)**

Use the Check command with the following data to determine that the domain is not available:

Domain Name: **example.info**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
domain:name avail='0'
```

### 2.3.1.21 Query Domain

Supply the following information to the Info command.

Domain Name: **example.info**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

```
Domain Name: example.info
Client ID: ClientX
Domain Status: ok
Domain Contact (Registrant) ID: OTE-C1
Domain Admin Contact: OTE-C2
Domain Billing Contact: OTE-C3
Domain Technical Contact: OTE-C4
Domain Name Server: ns1.example.com
Domain Name Server: ns2.example.com
Auth Info: my secret
Created By: ClientX
Created Date: 2010-06-21T22:00:00.0Z
Expiration Date: 2012-06-21T22:00:00.0Z
Last Updated By: ClientX
```

### 2.3.1.22 Check Name Server (Available)

Supply the following to the Check command.

ID: Host Name: **ns1.example.info**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg><host:name
avail='1'>
```

### 2.3.1.23 Create Name Server

Supply the following to the Create command:

Host Name: **ns1.example.info**  
Host Address: **192.168.10.11**

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg>

#### 2.3.1.24 Check Name Server (Unavailable)

Supply the following to the Check command.

ID: Host Name: **ns1.example.info**

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg><host:name avail='0'>

#### 2.3.1.25 Query Name Server

Supply the following to the Host Info command.

Host Name: **ns1.example.info**

Verify that the following response is received:

Host Name: **ns1.example.info**  
Client ID: **ClientX**  
Host IP Address: **192.168.10.11**  
Created By: **ClientX**  
Created Date: **2010-06-21T22:00:00.0Z**  
Client Trans ID: **11AA**  
Server Trans ID: **22BB**  
Status: **ok**

#### 2.3.1.26 Check Name Server (Available)

Supply the following to the Check command.

ID: Host Name: **ns2.example.info**

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg><host:name avail='1'>

#### 2.3.1.27 Create Name Server

Supply the following to the Create command:

Host Name: **ns2.example.info**

Host Address: **192.168.10.12**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### **2.3.1.28 Update Name Server (Add IP Address)**

Supply the following information to the Update command.

Host Name: **ns2.example.info**  
Add IP Address: **192.168.12.13**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### **2.3.1.29 Update Name Server (Remove IP Address)**

Supply the following information to the Update command.

Host Name: **ns2.example.info**  
Remove IP Address: **192.168.12.13**

Verify that the following response is received.

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### **2.3.1.30 Check Domain (Domain Available for Registration)**

Use the Check command with the following data to determine that the domain is available:

Domain Name: **domain.info**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>  
domain:name avail='1'
```

#### **2.3.1.31 Create Domain domain.info**

Create a new domain and associate two (2) Name Servers and four (4) Contacts to it by supplying the following elements to the Create command.

Domain Name: **domain.info**  
Domain Server: **ns1.example.info**  
Domain Server: **ns2.example.info**

Domain Registrant Contact ID: **OTE-C1**  
Domain Admin Contact ID: **OTE-C2**  
Domain Billing Contact ID: **OTE-C3**  
Domain Technical Contact ID: **OTE-C4**  
Domain Registration Period (Year): **1**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.1.32 Query Domain

Supply the following information to the Info command.

Domain Name: **domain.info**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

Domain Name: **domain.info**  
Client ID: **ClientX**  
Domain Status: **ok**  
Domain Contact (Registrant) ID: **OTE-C1**  
Domain Admin Contact: **OTE-C2**  
Domain Billing Contact: **OTE-C3**  
Domain Technical Contact: **OTE-C4**  
Domain Name Server: **ns1.example.info**  
Domain Name Server: **ns2.example.info**  
Auth Info: **my secret**  
Created By: **ClientX**  
Created Date: **2010-06-21T22:00:00.0Z**  
Expiration Date: **2011-06-21T22:00:00.0Z**  
Last Updated By: **ClientX**

### 2.3.1.33 Renew Domain

First, get the Expiration Date of the domain by issuing the Info command with the following data.

Domain Name: **domain.info**

Examine the Expiration Date returned from the previous command (output should be similar to the following).

Domain Expiration Date: **2011-06-21T22:00:00.0Z**

Issue the Renew command with the following data.

Domain Name: **domain.info**

Current Expiration Date: **2010-06-21** (returned in the previous Info command)  
Domain Years Period: **3**

Verify the output so that the expected Expiration Date is correct.

Domain Name: **domain.info**  
Expiration Date: **2014-06-21T22:00:00.OZ**

#### **2.3.1.34 Update Domain – Change Name Servers**

Enter the following information to the Update command.

Domain Name: **domain.info**  
Remove Name Server: **ns1.example.info**  
Remove Name Server: **ns2.example.info**

Add Name Server: **ns1.example.com**  
Add Name Server: **ns2.example.com**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### **2.3.1.35 Update Domain - Change Contact**

Issue the Update command with the following data: Remove the Contact, **OTE-C2** from Admin Contact and add the contact **OTE-C4** as a new Admin Contact.

Domain Name: **domain.info**  
Remove Admin Contact ID: **OTE-C2**  
Add Admin Contact ID: **OTE-C4**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### **2.3.1.36 Update Domain – Change Authorization Information**

Change authorization information of a domain by issuing the Update command with the following values.

Domain Name: **domain.info**  
New Auth Info: **new secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### **2.3.1.37 Update Domain - Change Domain Status**

Change the status of a domain by issuing the Update command with the following values.

Domain Name: **domain.info**  
Add Domain Status: **clientUpdateProhibited**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

## 2.3.2 Transfer of Objects

The following tests exercise EPP commands that revolve around object transfers.

### 2.3.2.1 Contact Transfer Request

This section tests the client's ability to request the transfer of a contact owned by another Registrar, **ClientY**. Please note that the Contact, **OTE-C5**, for which the transfer has been requested, was seeded in the Test Registry by AFILIAS Technical Support prior to the start of the test. Supply the following information to the Transfer command with the op='request' attribute and the following information.

ID: **OTE-C5**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.2.2 Query Contact Transfer

Use the Transfer command's op='query' attribute, along with the following information.

ID: **OTE-C5**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg> Contact  
Transfer Status: pending
```

### 2.3.2.3 Approve Contact Transfer

Another Registrar, **ClientY**, has an outstanding Transfer Request of one of **ClientX**'s Contacts, **OTE-C6**. This section involves the approval of the transfer request. Supply the following information to the Transfer command with the op='approve' attribute.

ID: **OTE-C6**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### 2.3.2.4 Reject Contact Transfer

Another Registrar, **ClientY**, has an outstanding Transfer Request of one of **ClientX**'s Contacts, **OTE-C7**. This section involves the rejection of the transfer request. Supply the following information to the Transfer command with the op='reject' attribute.

ID: **OTE-C7**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### 2.3.2.5 Domain Transfer Request

This section tests the client's ability to request the transfer of a domain, **transfer3.info**, owned by another Registrar, **ClientY**. Please note that the domain, **transfer3.info**, for which the transfer has been requested, was seeded in the Test Registry by AFILIAS Technical Support prior to the start of the test. Supply the following information to the Transfer command with the op='request' attribute and the following information.

Domain Name: **transfer3.info**  
Auth Info: **my secretY**

Verify that the following response is received:

```
<result code='1001'><msg lang='en-US'>Command completed successfully</msg>
```

#### 2.3.2.6 Approve Domain Transfer

Another Registrar, **ClientY**, has made a transfer request for one of **ClientX**'s domains, **transfer2.info**. This section involves the approval of this transfer request. Check the status of the transfer using the Transfer command with the op='query' attribute and the following information:

Domain Name: **transfer2.info**  
Auth Info: **my secretX**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>  
Transfer Status: pending
```

Approve this transfer by using the Transfer command with the op='approve' attribute and the following

information:

Domain Name: **transfer2.info**  
Auth Info: **my secretX**

Verify the following output:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.2.7 Reject Domain Transfer

Another Registrar, **ClientY**, has made a transfer request for one of **ClientX**'s domains, **transfer1.info**. This section involves the rejection of this transfer request. Reject the transfer by using the Transfer command with the op='reject' attribute and the following information:

Domain Name: **transfer1.info**  
Auth Info: **my secretX**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

## 2.3.3 Client Error Handling

The following section exercises the client's ability to correctly handle common EPP exceptions. The client should remain connected to the Test Registry despite the receipt of exceptions. A definition of each exception code is provided.

### 2.3.3.1 Correctly Handle 2003 Exception

2003 "Required parameter missing" - This response code must be returned when a server receives a command for which a required parameter value has not been provided. Submit the following using the Create command (do NOT submit a value for auth info):

Domain Name: **exception.info**  
Domain Server: **ns1.example.info**  
Domain Server: **ns2.example.info**  
Domain Registrant Contact ID: **OTE-C1**  
Domain Admin Contact ID: **OTE-C2**  
Domain Billing Contact ID: **OTE-C3**  
Domain Technical Contact ID: **OTE-C4**

Verify that the following response is received:

```
<result code='2003'><msg lang='en-US'>Required parameter missing</msg>
```

**Note:** This error is due to the fact that domain auth info was not provided in the create command.

### 2.3.3.2 Correctly Handle 2005 Exception

2005 "Parameter value syntax error" - This response code must be returned when a server receives a command containing a parameter whose value is improperly formed. The error value should be returned via an element in the EPP response.

Submit the following using the Create command:

```
Domain Name: -*invalid.info  
Domain Server: ns1.example.info  
Domain Server: ns2.example.info  
Domain Registrant Contact ID: OTE-C1  
Domain Admin Contact ID: OTE-C2  
Domain Billing Contact ID: OTE-C3  
Domain Technical Contact ID: OTE-C4  
Auth Info: my secret
```

Verify that the following response is received:

```
<result code='2005'><msg lang='en-US'>Parameter value syntax error</msg>
```

**Note:** This error is due to the fact that domain name starts with invalid character.

### 2.3.3.3 Correctly Handle 2306 Exception

2306 "Parameter value policy error" - This response code must be returned when a server receives a command containing a parameter value that is syntactically valid, but semantically invalid due to local policy. For example, the server *may* support a subset of a range of valid protocol parameter values. The error value should be returned via an element in the EPP response.

Submit the following using the Create command:

```
Domain Name: exception.info  
Domain Server: ns1.example.info  
Domain Server: ns2.example.info  
Domain Registrant Contact ID: OTE-C1  
Domain Admin Contact ID: OTE-C2  
Domain Billing Contact ID: OTE-C3  
Domain Technical Contact ID: OTE-C4  
Domain Period (Years): 99  
Auth Info: my secret
```

Verify that the following response is received:

```
<result code='2306'><msg lang='en-US'>Parameter value policy error</msg>
```

**Note:** This is due to the fact that value entered for Domain Period is outside valid range (1 to 10 years).

### 2.3.3.4 Correctly Handle 2002 Exception

2002 "Command use error" – This response code must be returned when a server receives a command

that is properly formed, but cannot be executed due to a sequencing or context error.

Submit the following using the Renew command:

Domain Name: **example.info**  
Expiration Date: **2011-06-21**

Verify that the following response is received:

```
<result code='2306'><msg lang='en-US'>Command use error</msg>
```

**Note:** This is due to the fact that a wrong Expiration date was entered (the correct Expiration date is 2012-06-21) while renewing the domain.

### 2.3.3.5 Correctly Handle 2303 Exception

2303 "Object does not exist" - This response code must be returned when a server receives a command to transform an object that does not exist in the registry.

Submit the following using the Create command:

Domain Name: **exception.info**  
Domain Server: **ns1.example.info**  
Domain Server: **ns2.example.info**  
Domain Registrant Contact ID: **OTE-C99**  
Domain Admin Contact ID: **OTE-C2**  
Domain Billing Contact ID: **OTE-C3**  
Domain Technical Contact ID: **OTE-C4**  
Domain Period (Years): **2**  
Auth Info: **my secret**

Verify that the following response is received:

```
<result code='2303'><msg lang='en-US'>Object does not exist</msg>
```

**Note:** Registrant Contact ID, **OTE-C99**, does not exist in the registry and hence the error.

### 2.3.3.6 Correctly Handle 2305 Exception

2305 "Object association prohibits operation" - This response code must be returned when a server receives a command to transform an object that cannot be completed due to dependencies on other objects that are associated with the target object. For example, a server *may* disallow commands while an object has active associations with other objects.

Submit the following to the Delete command:

Contact ID: **OTE-C2**

Verify that the following response is received:

```
<result code='2305'><msg lang='en-US'>Object association prohibits operation</msg>
```

**Note:** The error is due to the fact that the contact, **OPE-C2**, is associated with multiple domains.

### 2.3.3.7 Correctly Handle 2201 Exception

2201 "Authorization error" - This response code must be returned when a server notes a client authorization error when executing a command. This error is used to note that a client lacks privileges to execute the requested command.

Submit the following to the Delete command:

Domain Name: **transfer3.info**

Verify that the following response is received:

```
<result code='2201'><msg lang='en-US'>Authorization error</msg>
```

**Note:** The error is due to the fact that the domain, **transfer3.info**, is not owned by registrar, **ClientX**.

## 2.4 IDN EPP Acceptance Criteria (Optional)

This is an optional OT&E test for Registrars who wish to implement EPP for International Domain Name (IDN) operations with the Registry. Registrars do not have to pass this test to begin IDN registrations.

If a registrar has already passed the regular EPP OT&E test and would like to complete the optional IDN test at a later stage, then Steps **2.3.1.1** to **2.3.1.10** and **2.3.1.14** to **2.3.1.17** under section **2.3.1** will have to be completed first to create objects which will be used in steps under sections **2.4.1** and **2.4.2**.

If a registrar is taking the regular EPP OT&E test and optional IDN test together then all steps listed under both "**2.3 Mandatory EPP Acceptance Criteria**" and "**2.4 IDN EPP Acceptance Criteria (Optional)**" will need to be completed, except the steps **2.4.3.2** to **2.4.3.7**, listed under "**2.4.3 Delete IDN And Other Objects**", which the registrar will **NOT** need to perform, as those objects are deleted in sections "**2.5 Deletion of Other Objects**" in steps **2.5.3** to **2.5.8** as part of EPP acceptance test.

### 2.4.1 Creation of IDN objects and their updates

#### 2.4.1.1 Check Domain (Domain Available for Registration)

Use the Check command with the following data to determine that the Hungarian IDN domain is available:

Domain Name: **xn--abcxyz-rta.info**  
IDN Script: **hu**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg> domain:name
```

avail='1'

#### 2.4.1.2 Create IDN Domain using the Hungarian script

Domain Name: **xn--abcxyz-rta.info**  
Domain Server: **ns1.example.com**  
Domain Server: **ns2.example.com**  
Domain registrant contact: **OTE-C1**  
Domain Admin contact: **OTE-C2**  
Domain Billing contact: **OTE-C3**  
Domain Technical contact: **OTE-C4**  
Auth Info: **my secret**  
IDN Script: **hu**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### 2.4.1.3 Check Domain (Domain Not Available for Registration)

Use the Check command with the following data to determine if the domain is available:

Domain Name: **xn--abcxyz-rta.info**  
IDN Script: **hu**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg> domain:name  
avail='0'
```

#### 2.4.1.4 Query Hungarian IDN Domain

Domain Name: **xn--abcxyz-rta.info**

Verify that the following response is received:

Domain Name: **xn--abcxyz-rta.info**  
Client ID: **ClientX**  
Domain Status: **ok**  
IDN Script: **hu**  
Domain Server: **ns1.example.com**  
Domain Server: **ns2.example.com**  
Domain Contact (Registrant): **OTE-C1**  
Domain Admin Contact: **OTE-C2**  
Domain Billing Contact: **OTE-C3**  
Domain Technical Contact: **OTE-C4**  
Auth Info: **my secret**

Created By: **ClientX**  
Created Date: **2010-06-22**  
Expiration Date: **2012-06-22**  
Last Updated By: **ClientX**

*Note: The EPP response should contain the <idn:infData> and <idn:script> tags.*

#### **2.4.1.5 Update Domain with Supported IDN Script**

Query the domain name: **xn--abcxyz-rta.info**

Verify that the following response is received:  
Domain name: **xn--abcxyz-rta.info**  
IDN script: **hu**

Issue the domain update command with the following:  
Domain name: **xn--abcxyz-rta.info**  
IDN Script: **is**

Verify that the following response is received:

Command Completed Successfully  
Domain Name: **xn--abcxyz-rta.info**  
IDN Script: **is**

#### **2.4.1.6 Check Multiple IDN with proper Script**

Domain Name: **xn--abcxyz-rta.info**  
Domain Name: **xn--abcabc-rta.info**  
IDN Script: **is**

Verify that the following response is received:

Command Completed Successfully  
Domain Name Result: **0**  
Domain Name Result: **1**

#### **2.4.1.7 Check IDN with invalid script**

Submit the following on Domain Check command:  
Domain Name: **xn--dn-mja.info**  
IDN Script: **sv**

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg> domain:name avail='0'

*Note: While checking an IDN with characters that do not appear in the script table of script submitted, the server response is to tell the client that the domain is not available.*

## **2.4.2 Client Error Handling on IDN**

### **2.4.2.1 Correctly Handle 2003 Error Exception**

Create an IDN as a regular domain. Submit the following to the domain create command:

Domain Name: **xn--bq-uia.info**  
Domain Server: **ns1.example.com**  
Domain Server: **ns2.example.com**  
Domain registrant contact: **OTE-C1**  
Domain Admin contact: **OTE-C2**  
Domain Billing contact: **OTE-C3**  
Domain Technical contact: **OTE-C4**  
Auth Info: **my secret**

Verify that the following response is received:

2003:Required parameter missing (idn:create)

*Note: Since this domain is being created as a regular domain, the XML should not contain any of the <IDN> tags within the <UNSPEC> area.*

### **2.4.2.2 Correctly Handle 2003 Error Exception**

Check IDN as regular domain

Domain Name: **xn--bq-uia.info**

Verify that the following response is received:

2003:Required parameter missing

Note: Checking for an IDN, but using XML for a regular domain:check command.

### **2.4.2.3 Correctly Handle 2306 Error Exception**

Create IDN with empty script parameter

Domain Name: **xn--bq-uia.info**  
Domain Server: **ns1.example.com**  
Domain Server: **ns2.example.com**  
Domain registrant contact: **OTE-C1**  
Domain Admin contact: **OTE-C2**  
Domain Billing contact: **OTE-C3**

Domain Technical contact: **OTE-C4**  
Auth Info: **my secret**

Verify that the following response is received:

2306:Parameter value policy error

Note: This domain is being created as an IDN, but no script is specified.

#### 2.4.2.4 Correctly Handle 2306 Error Exception

Create an IDN with invalid script name

Domain Name: **xn--bq-uia.info**  
Domain Server: **ns1.example.com**  
Domain Server: **ns2.example.com**  
Domain registrant contact: **OTE-C1**  
Domain Admin contact: **OTE-C2**  
Domain Billing contact: **OTE-C3**  
Domain Technical contact: **OTE-C4**  
Auth Info: **my secret**  
IDN Script: **hu-AT**

Verify that the following response is received:

2306:Parameter value policy error

Note: This domain is being created using a script (hu-AT) that does not exist.

### 2.4.3 Delete IDN And Other Objects

#### 2.4.3.1 Delete IDN Domain

Delete the Domain, **xn--abcxyz-rta.info**, created in Section 2.4.1.2, by supplying the following information to the Domain Delete command.

Domain Name: **xn--abcxyz-rta.info**

Verify that the following response is received:

<result code='1000'><msg lang='en-US'>Command completed successfully</msg>

**Please note that the following contact delete operations, under 2.4.3.2 to 2.4.3.5 can be completed if only the optional IDN test is taken. If the registrar is also taking mandatory EPP acceptance criteria test along with the optional IDN test then these steps, ie. 2.4.3.2 to 2.4.3.5, will give error as the contact objects, OTE-C1 to OTE-C4, are already associated to domains, “example.info” and “domain.info”, created in steps, 2.3.1.19 and 2.3.1.31, as part of the**

mandatory EPP acceptance test.

#### 2.4.3.2 Delete Contact (OTE-C1)

This step is same as in “2.5.3 Delete Contact (OTE-C1)”.

#### 2.4.3.3 Delete Contact (OTE-C2)

This step is same as in “2.5.4 Delete Contact (OTE-C2)”.

#### 2.4.3.4 Delete Contact (OTE-C3)

This step is same as in “2.5.5 Delete Contact (OTE-C3)”.

#### 2.4.3.5 Delete Contact (OTE-C4)

This step is same as in “2.5.6 Delete Contact (OTE-C4)”.

#### 2.4.3.6 Delete Name Server (ns1.example.com)

This step is same as in “2.5.7 Delete Name Server (ns1.example.com)”.

#### 2.4.3.7 Delete Name Server (ns2.example.com)

This step is same as in “2.5.8 Delete Name Server (ns2.example.com)”.

## 2.5 Deletion of Other Objects

The following tests exercise EPP commands that delete objects created during section 2.3.1.

### 2.5.1 Delete Domain (example.info)

Delete the Domain, **example.info**, created in Section 2.3.1.19 by supplying the following information to the Domain Delete command.

Domain Name: **example.info**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

**Note:** Deletion of parent domain, **example.info**, also deletes child hosts, **ns1.example.info** and **ns2.example.info**, created in 2.3.1.23 and 2.3.1.27, as they are not associated with any other domains.

### 2.5.2 Delete Domain (domain.info)

Delete the Domain, **domain.info**, created in Section 2.3.31 by supplying the following information to the

Domain Delete command.

Domain Name: **domain.info**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### **2.5.3 Delete Contact (OTE-C1)**

Delete the Contact, **OTE-C1**, created in Section **2.3.1.2** by supplying the following information to the Contact Delete command.

Contact ID: **OTE-C1**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### **2.5.4 Delete Contact (OTE-C2)**

Delete the Contact, **OTE-C2**, created in Section **2.3.1.6** by supplying the following information to the Contact Delete command.

Contact ID: **OTE-C2**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### **2.5.5 Delete Contact (OTE-C3)**

Delete the Contact, **OTE-C3**, created in Section **2.3.1.8** by supplying the following information to the Contact Delete command.

Contact ID: **OTE-C3**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### **2.5.6 Delete Contact (OTE-C4)**

Delete the Contact, **OTE-C4**, created in Section **2.3.1.10** by supplying the following information to the Contact Delete command.

Contact ID: **OTE-C4**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.5.7 Delete Name Server (ns1.example.com)

Delete the name server, **ns1.example.com**, created in section 2.3.1.15, by supplying the following information to Host Delete command.

Host Name: **ns1.example.com**

Verify that the following response is received.

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.5.8 Delete Name Server (ns2.example.com)

Delete the name server, **ns2.example.com**, created in section 2.3.1.17, by supplying the following information to Host Delete command.

Host Name: **ns2.example.com**

Verify that the following response is received.

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

## 2.6 Efficiency of Client Session Management

This section exercises the client's ability to utilize commands that must be represented as empty elements, with no child objects.

### 2.6.1 Keep Session Alive

For this test, the client must keep the current session open to the Registry for at least 30 minutes before proceeding to the next section. Use the Hello command at intervals under 10 minutes to maintain client connectivity.

### 2.6.2 Request Message Queue Information

Clients may use the poll command to retrieve messages queued by the server. Issue the poll command with the op='request' attribute to retrieve queue information, and the first message within the queue.

Verify that the following response is received:

```
<response><result code='1301'><msg lang='en-US'>Command completed successfully; ack to  
dequeue</msg></result><msgQ count='48' id='43'><msg lang='en-US'>Transfer  
Requested.</msg>
```

**Note:** The value returned for 'id' will be necessary for section 2.7.3

### 2.6.3 Ack Queued Message

Issue the poll command with the op=ack attribute to acknowledge receipt of the first message, and remove it from the queue.

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg></result><msgQ count='47' id='45'>
```

## 2.7 End Session

For a Registrar client to end communications with the Registry, the Logout command is used with no arguments.

If successful, the Registry will send the following response and then end the session.

```
<result code='1500'><msg lang='en-US'>Command completed successfully; ending  
session</msg>
```

## 2.8 Completing the Test

At this point, contact AFILIAS Technical Support by logging to the communication portal by using your Registrar Relations login credentials at [https://registrars.afilias.info/cgi-bin/ssl/open\\_update\\_ticket.cgi](https://registrars.afilias.info/cgi-bin/ssl/open_update_ticket.cgi) or call +1.4166463306 and inform them that you have completed this test.

## Appendix A - Seeded Registry information

The OT&E test requires the creation and manipulation of several EPP objects prior to the client's initial connection. Afilias Technical Support will perform the necessary operations before the client's initial connection. The data within this Appendix is included for informational purposes only.

**\*\*\* Registrar: Do not attempt to enter this data into the Test Registry. \*\*\***

### User

Registrar: **ClientX**  
Password: **foo-BAR2**

### Contacts

The Contact ID values for each of the seeded contacts are as follows:

Object	Owned By	Notes
<b>OTE-C5</b>	<b>ClientY</b>	
<b>OTE-C6</b>	<b>ClientX</b>	Auth Info: my secret ** This contact has pending transfer status, initiated by ClientY**
<b>OTE-C7</b>	<b>ClientX</b>	Auth Info: my secret ** This contact has pending transfer status, initiated by ClientY**

The seeded contacts use the following common values:

Contact Name: **Test Contact**  
Contact Organization: **Example Corp. Inc**  
Contact Address Street: **123 Example St.**  
Contact Address Street: **Suite 100**  
Contact Address City: **Anytown**  
Contact Address State/Province: **Any Prov**  
Contact Address Postal Code: **A1A1A1**  
Contact Address Country: **CA**  
Contact Voice: **+1.4165555555**  
Contact Voice Extension: **1111**  
Contact Fax: **+1.4165555556**  
Contact Email: [jdoe@test.test](mailto:jdoe@test.test)

The seeded hosts are as follows:

Object	Owned By	Notes
<b>ns1.sample.com</b>	<b>ClientX</b>	
<b>ns2.sample.com</b>	<b>ClientX</b>	

### Seeded Domains:

<b>Object</b>	<b>Owned By</b>	<b>Notes</b>
<b>transfer3.info</b>	<b>ClientY</b>	Auth Info: <b>my secretY</b> , <b>OPE-C5</b> for all contact types
<b>transfer1.info</b>	<b>ClientX</b>	Auth Info: <b>my secretX</b> , <b>OPE-C6</b> for all contact types ** This domain has pending transfer status, initiated by ClientY**
<b>transfer2.info</b>	<b>ClientX</b>	Auth Info: <b>my secretX</b> , <b>OPE-C6</b> for all contact types ** This domain has pending transfer status, initiated by ClientY**

All seeded domains above use seeded name server values: **ns1.sample.com** and **ns2.sample.com**.