Local Certificates

This document describes how local loopback IP addresses are assigned to Neon Research related web sites and APIs for development and testing purposes and how these applications are secured via SSL/TLS. This security is required due to our reliance on OAuth/OpenID for identity management.

# Loopback Addresses

TCP/IP specifies that packet sent to IP addresses in the range of 127.0.0.1 - 127.255.255.255 are not to be routed across the internet, but are instead to be terminated on the same network device that received the packet for transmission. This is a convenient way to send network traffic to the local machine.

# .INFO Domain Convention

Neon Research reserves domains under the **.info** top-level domain for **local development** and testing purposes only. For example, **neonbowl.info** is registered for testing the bowling related sites and services. Other TLDs such as **.com**, **.net**, and **.io** are all reserved for production purposes.

Separating development domains from production domains makes it convenient to create separate SSL certificates for each where the .info certificates and private keys may be stored in the source code repository with relatively lower security but much better deployment convenience for developers and production certificates and keys would be stored elsewhere.

# Local Service Hosts

Neon Research uses loopback addresses to make it easy for developers to locally deploy and test applications locally on their workstations, even for network services that require SSL/TLS security.   
  
The approach for each application is to:

1. Assign one or more FQDNs to the application based on the application’s main domain: e.g. **api.neonbowl.com**. This will be included in the endpoint URIs used by client and services that consume the application and will typically be resolved by a global traffic manger enabled DNS service (such as Akamai GTM).
2. A FQDN with the same host name and the **\*.info** top-level domain (such as **api.neonbowl.info**) will be registered with a static DNS service. This name will be assigned an **A record** with the loopback address **127.127.0.0** (chosen to avoid conflicts with the Azure and Service Fabric emulators) and will be used to address the service locally on developer workstations.  
     
   Note: All **\*.info** endpoints **share the same loopback address**. This works because Neon Research requires that all HTTP clients consuming the service over TLS support Subject Name Indication (SNI)
3. Create the private key and SSL certificate for the host and add it to the encrypted  
    **$\Certs\Local Certificates.zip** file within the source code repository. Certificates keys and files will be named for the fully qualified host: e.g. **local-api.test.com.key** and **local-api.test.com.cer**.

Password Hint: **What does Spot do?**  
  
**IMPORTANT: Only local/development test certificates should be stored here. Production certificates must be secured elsewhere.**

See the instructions below for the steps required to generate a private key and certificate.

1. The SSL Certificates must be installed on the local machine. The **$\Certs\install-certs.cmd** batch file handles the installation of all development certificates. You’ll need to pass the password used to encrypt the certificate archive file.

# Application Host Names

The table below describes the current test domain and service host names. This table should be kept up to data as application hosts are added and created. You should also update the **ssl-config.cmd** and   
**ssl-unconfig.cmd** files to ensure the development environments will have the required HTTP certificates and ACLs configured (by editing the HOSTS variable in the scripts).

|  |  |
| --- | --- |
| Domain/Host | Description |
|  |  |
| neonbowl.info | Bowling applications and services |
| Issuer: | CheapDomain.com (uid: jslill) |
| Type: | wildcard |
| Expires: | 5/22/2016 |
|  |  |
| @, www | NeonBowl website |
| api | NeonBowl WebAPI |
| sts | Secure Token Service (IdentityServer3) |
| mdb | Mongo database |
| mdr | Mongo router |
| cops | Cassandra operations console |
| cdb | Cassandra database |
| esr | Elasticsearch Router/Kibana |
| esd | Elasticsearch database |
| Rds | Redis node |

# Generating an .INFO Private Key and Certificate

Follow the steps below to generate a private key and SSL certificate.

1. Invoke the **openssl-csr.cmd** batch file (located on the PATH in **$\ToolBin\OpenSSL**), passing the **FQDN** of the domain being secured:  
     
   openssl-csr mydomain.info  
     
   You will be prompted for additional information. The defaults come from the **$\ToolBin\OpenSSL\openssl.cnf** file. The **Organization Name** defaults to **Neon Research**. You may want to customize the **Organization Unit** to describe the specific project or business (e.g. **NeonBowl**). You will also need to enter the **wild-carded Site FQDN** (e.g: **\*.mydomain.info**)
2. This will generate the private **\*.key** file as well as the certificate signing request **\*.csr** file in the **%NR\_TEMP%\OpenSSL** folder.
3. Go to your certificate authority’s website (<http://CheapDomain.com> [uid: jslill] at this time) and create a **wildcard certificate**.
4. Open the **\*.csr** file in a **text editor** and copy/**paste** the contents into the **CSR text box** on the issuer’s site and complete the steps to **create** the certificate.
5. **Download** the certificate **ZIP** file (targeting **IIS**) from the issuer. **Extract** the contents of the ZIP file into temporary folder where the key and CSR files were created. Rename the **\*.crt** file to **<domain>.crt**.
6. Open the **$\Certs\Local Certificates.7z** file and **create a folder** named for the domain (e.g. neonbowl.info).
7. Create a **\*.pfx certificate** by combining the private key with the \*.crt certificate from the issuer:  
   1. Open a **DOS** command window (run **as Administrator**).
   2. Change the **current directory** to the folder holding the private key and CRT files.
   3. Run this command, passing the domain: openssl-pfx . <domain>
   4. Enter the same **password** used to encrypt the **Local Certificates.7z file** twice. The **\*.pfx certificate** should have been created in the current folder.
8. **Copy** the all the files from the temporary certificates folder into the new archive folder. Verify that you have **\*.key**, **\*.csr**, and **\*.pfx** files and then **close 7z**.
9. **Update** the certificate issuer and expiration **information** in the **Domain and Host Mappings** table above.
10. Follow the **Install Local SSL Certificates** instructions in **$\Setup-Common.docx**.
11. Run **mmc.exe** and expand **Certificates (Local Computer)\Personal**. Click on **Certificates** and press **F5** to refresh. You should see the new wildcard certificate in the right pane.
12. **Double-click** the certificate and select the **Details** tab and select **Thumbprint**. Select the HEX characters in the bottom pane and then **CTRL-C** to copy the text into the clipboard.
13. Open **$\Certs\config-certs.cmd** in Notepad and then:  
    1. Add or update the **thumbprint definition** for the domain by pasting it from the clipboard) and then **removing** any **spaces**.
    2. For new domains: Add a **loop sections** at the bottom that configures the certificate bindings and ACLs. Open **$\Certs\unconfig-certs.cmd** and add loop sections to revert the domain bindings.
14. Verify that **$\Certs\Local Certificates.7z** is still password protected before committing any changes to the repository.
15. **Commit** all changes to the source repository.
16. **Delete** all temporary folders holding the **private key.**