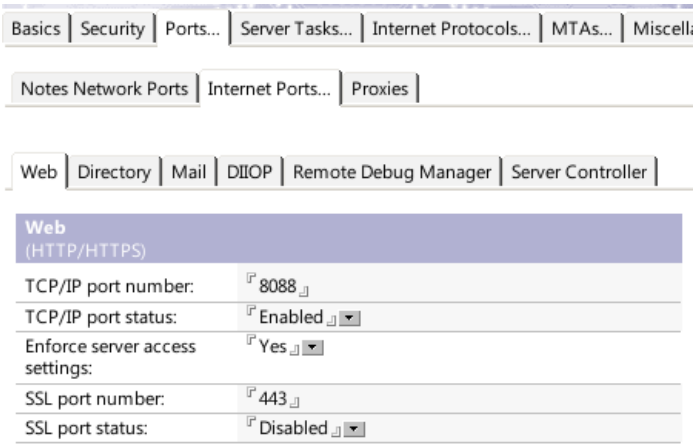
Setting up nginx in Front of a Domino Server

18/09/2014

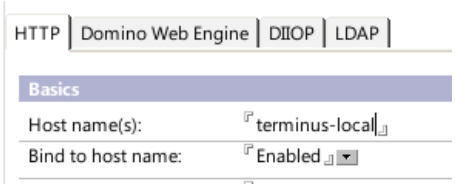
As I've mentioned before and now presented on, I'm a big proponent of using a reverse proxy in front of Domino. There are numerous benefits to be gained, particularly when you expand your infrastructure to include multiple back-end servers. But even in the case of a single server, I've found it very worthwhile to set up, and not overly complicated. This example uses nginx and Domino on Ubuntu Linux, but the ideas and some configuration apply much the same way on other OSes and with other web servers.

Domino

The first step involves a bit of configuation on the Domino server. The first is to move Domino off the main port 80, disable SSL, and, ideally, bind it to a local-only IP address. The port setting is familiar - I picked port 8088 here, but it doesn't matter too much what you pick as long as it doesn't conflict with anything else on your server:



The next step is to bind Domino to a local-only adapter so external clients don't access its HTTP stack directly. In this example, I have a LAN-only adapter whose IP address I named "terminus-local" in /etc/hosts, but I imagine "localhost" would work just fine in this case:



Once that's set, the last stage of configuration is to enable the WebSphere connector headers by setting a notes.ini property:

```
HTTPEnableConnectorHeaders=1
```

Enabling these will allow us to send specialized headers from our reverse proxy to Domino to make Domino act as if the request is coming to it directly.

After that, restart Domino (or just HTTP, probably).

nginx

Next, it's on to setting up nginx. On Ubuntu/Debian, it's pretty straightforward:

```
# apt-get install nginx
```

The main config file /etc/nginx/nginx.conf should be good as-is. The way the Ubuntu config works, you set up individual web site files inside the /etc/nginx/sites-available directory and then create symlinks to them in the /etc/nginx/sites-enabled directory. Out of convention, I name them like "000-somesite" to keep the priority clear. The first file to create is a site to listen on port 80, which will serve entirely as a redirect to SSL. You don't *have* to do this - instead, you could bring the content from the next file into this one instead of the redirection line. This is usually a good idea, though. This file is `001-http-redirect`:

```
server {
    listen [::]:80;

    return https://$host$request_uri;
}
```

The only really oddball thing here is the "listen" line. Normally, that would just be "listen 80", but adding the brackets and colons allows it to work on IPv4 and IPv6 on all addresses.

The next file is the important one for doing the proxying, as well as SSL. It's `002-domino-ssl`:

```
server {
        listen [::]:443;

        client_max_body_size 100m;

        ssl on;
        ssl_certificate /etc/nginx/ssl/ssl-unified-noanchor.pem;
        ssl_certificate_key /etc/nginx/ssl/ssl.key;

        location / {
                proxy_read_timeout 240;
                proxy_pass http://localhost:8088;
                proxy_redirect off;
                proxy_buffering off;

                proxy_set_header        Host                $host;
                proxy_set_header        X-Forwarded-For     $proxy_add_x_forwarded_for;
                proxy_set_header        $WSRA               $remote_addr;
                proxy_set_header        $WSRH               $remote_addr;
                proxy_set_header        $WSSN               $host;
                proxy_set_header        $WSIS               True;
        }
}
```

The `client_max_body_size` line is to allow uploads up to 100MB. One thing to be aware of when using proxies is that they can impose their own limits on request sizes just as Domino does, and nginx's default is relatively low.

nginx's keychain format is *almost* as simple as just pointing it to your certificate and private key, with one catch: to have intermediate signing certificates (like those from your SSL provider or registrar), you concatenate the certificates into a single file. This tutorial covers it (and this config) nicely.

The core of the reverse proxy comes in with that `location /` block. In a more-complicated setup, you might have several such blocks to point to different apps, app servers, or local directories, but in this case we're just passing everything directly through to Domino. The first four lines do just that, setting a couple options to account for very-long-loading pages, to point to Domino, and some other options.

The `proxy_set_header` lines are the payoff for the connector headers we set up in Domino. The first is to pass the correct host name on to Domino so it knows which web site document to use, the second is a fairly standard-outside-of-Domino header for reverse proxies, and then the rest are a set of the available WebSphere (hence "$WS") headers, specifying what Domino should see as the remote address, the remote host name (I don't have nginx configured to do reverse DNS lookups, so it's the same value), the host name again, and whether or not it should act as being over SSL.

Once that's set, create symlinks to these files in the `sites-enabled` directory from the `sites-available` directory and restart nginx:

```
# ln -s ../sites-enabled/001-http-redirect
# ln -s ../sites-enabled/002-domino-ssl
# service nginx restart
```

Assuming all went well, you should be all set! This gets you a basic one-server proxy setup. The main advantage is the superior SSL handling - nginx's SSL stack is OpenSSL and thus supports all the modern features you'd expect, including SHA-2 certificates and the ability to serve up multiple distinct SSL certificates from the same IP address (this would be done with additional config files using the `server_name` parameter after `listen`). Once you have this basis, it's easy to expand into additional features: multiple back-end servers for load balancing and failover, better error messages when Domino crashes (which is more frequent than nginx crashing), and nifty plugins like GeoIP and mod_pagespeed.

3 Comments

Ray Bilyk **18/09/2014**

Your presentation was great! I just wished I didn't miss the beginning of it. I hope to see it again soon... hopefully in the January timeframe...

Richard Moy **18/09/2014**

Great Job Jesse.

Tinus Riyanto **19/09/2014**

I am curios to know if it is possible for nginx to authenticate first all request before passing it to Domino. I assume this would involve setting up some sort of landing page on nginx and a way to "pass" authentication to Domino and reading its result ?

Post New Comment

Name

Email Address

URL

Body

Skrifttype Større…

Tryk på ALT 0 for at få hjælp

Post Comment