NGINX

Using NGINX Plus to Load Balance Microsoft® Exchange Servers

Deployment Guide
for NGINX Plus R6
and Microsoft® Exchange 2013

Published May 20, 2015

Table of Contents

This guide explains how to deploy NGINX Plus to load balance traffic across a pool of Microsoft Exchange servers.

You can deploy Exchange and NGINX Plus on premise, in a private cloud, or in public clouds including Amazon Web Services (AWS) and Microsoft Azure. The guide covers the different installation types, and provides complete instructions for customizing both NGINX Plus and Exchange as required.

About NGINX Plus

NGINX Plus is the commercially supported version of the open source NGINX software. NGINX Plus is a complete application delivery platform, extending the power of NGINX with a host of enterprise-ready capabilities that are instrumental to building web applications at scale::

- Full-featured HTTP and TCP load balancing
- High-performance reverse proxy
- Caching and offload of dynamic and static content
- Adaptive streaming to deliver audio and video to any device
- Application-aware health checks and high availability
- Advanced activity monitoring available via a dashboard or API
- Management and real-time configuration changes with DevOps-friendly tools

NGINX Plus Release 6 (R6) and later provides full-featured load balancing of TCP traffic as well as HTTP and HTTPS, making it ideal for Microsoft Exchange deployments, which use all three protocols.

Using this Guide

After reviewing the "Prerequisites and System Requirements" on **page 5**, perform the instructions in these sections:

- Configuring DNS, Exchange, and Firewalls on **page 6**
- Configuring NGINX Plus for Exchange on **page 12**

When configuring NGINX Plus, you choose between a basic deployment of load-balanced Exchange servers or a deployment with enhanced reliability, application health checks, and live activity monitoring. For details, see “Configuring NGINX Plus for Exchange“ on **page 12**.

About Sample Values and Copying of Text

- `company.com` is used as a sample domain name (mostly in key names and DNS entries). Replace it with your domain name.
- Many NGINX Plus configuration blocks in this guide list two sample client access servers (CASs) with IP addresses `10.0.0.237` and `10.0.0.238`. Replace these addresses with the IP addresses of your CASs. Include a line in the configuration block for each CAS if you have more or fewer than two. In contrast, port numbers are obligatory values except where noted.
- For readability reasons, some commands appear on multiple lines. If you want to copy and paste them into a terminal window, we recommend that you first copy them into a text editor, where you can substitute the correct object names for your deployment and remove any extraneous formatting characters that your PDF viewer might insert.
- The configuration examples in the step-by-step instructions include hyperlinks to the NGINX reference documentation, for easy access to more information about the directives. (If a directive appears multiple times in a section, only the first occurrence is hyperlinked.) We recommend that you do not copy hyperlinked text (or any other text) from this PDF file into your configuration files, because it might include unwanted link text and does not include whitespace and other formatting that makes the configuration easy to read. For more information, see "Creating and Modifying Configuration Files" on **page 12**.

Prerequisites and System Requirements

- Microsoft Exchange 2013 or later, installed and configured on a system running Windows ® Server 2012 or later. The load balancing functionality is not supported for earlier versions of Microsoft Exchange.

  Exchange CASs must be configured for Basic authentication, as specified in this guide.
- Linux system to host NGINX Plus (in on-premise and private-cloud deployments). To avoid potential conflicts with other applications, we recommend you install NGINX Plus on a fresh system. For the list of Linux distributions supported by NGINX Plus, see **NGINX Plus Technical Specifications**.
- NGINX Plus R6 or later. Full TCP load-balancing capabilities are not available in earlier versions of NGINX Plus.

The instructions assume you have basic Linux system administration skills, including the following. Full instructions are not provided for these tasks.

- Installing Linux software from vendor-supplied packages
- Editing configuration files
- Copying files between a central administrative system and Linux servers
- Running basic commands to start and stop services
- Reading log files

Similarly, the instructions assume you have basic Windows system administration skills, including the following.

- Logging in to a system through Microsoft Remote Desktop
- Running PowerShell commands
- Restarting Internet Information Services (IIS) services

Configuring DNS, Exchange, and Firewalls

To prepare for the configuration of NGINX Plus, first perform the steps in the following sections:

- Configuring DNS below
- Configuring Exchange below
- Configuring Firewalls on page 8

Configuring DNS

Exchange requires the following Domain Name System (DNS) records for normal operation. Create or modify them as necessary.

- An `MX` record for mail delivery.

```
company.com.  300   MX     10 mail.company.com
```

- An `A` record for the main email server. Replace `X.X.X.X` with the public IP address of your NGINX Plus server.

```
mail.company.com.  60     A      X.X.X.X
```

- A `TXT` record for Sender Policy Framework (SPF). Replace `X.X.X.X` with the public IP address of your NGINX Plus server. For more information about SPF records, see Sender ID Framework SPF Record Wizard in the Microsoft documentation.

```
company.com. 300    TXT    ''v=spf1 mx a ip4:X.X.X.X/32 -all''
```

- An `SRV` record for the Autodiscover service.

```
_autodiscover._tcp.company.com. 60     SRV    1 10 443 mail.company.com
```

Configuring Exchange

Use Exchange Management Shell (PowerShell) to configure Exchange on each CAS. When running the `Set` command, you must always specify the CAS name and directory (together referred to as the *identity*) for the object being configured. You can either include the `-Identity` parameter on the `Set` command line, or type the identity at the prompt that appears if you don't include the `-Identity` parameter (as in the commands in this section).

To obtain an identity if you don't know it, run the `Get` command that corresponds to the `Set` command you need to run. Include the `fl` (formatted list) keyword on the `Get` command line to view the complete output from the command. For example, to obtain identity information for the `Set-OutlookAnywhere` command, run the following command:

```
C:\> Get-OutlookAnywhere | fl
```

Identities are case insensitive, and generally include spaces, parentheses, and backslashes, as in these examples for a CAS called **CAS01**.

```
CAS01\Rpc (Default Web Site)
CAS01\mapi (Default Web Site)
CAS01\Autodiscover (Default Web Site)
```

Repeat these commands on each CAS in your deployment:

1. Working on the CAS, log in to PowerShell under an account with administrative privileges.
2. Open the **Start** menu and run the **Exchange Management Shell,** which is a terminal window.
3. Configure the external hostname for Outlook Anywhere.

```
   C:\> Set-OutlookAnywhere -ExternalHostname mail.company.com
```

4. Configure Basic authentication for Outlook Anywhere.

```
   C:\> Set-OutlookAnywhere -ExternalClientsRequireSsl 1
-DefaultAuthenticationMethod basic
-ExternalClientAuthenticationMethod basic
-IISAuthenticationMethods basic
-InternalClientAuthenticationMethod basic
```

5. Configure Basic authentication for Autodiscover.

```
   C:\> Set-AutodiscoverVirtualDirectory
-LiveIdNegotiateAuthentication 0
-WSSecurityAuthentication 0 -LiveIdBasicAuthentication 0
-BasicAuthentication 1 -DigestAuthentication 0
-WindowsAuthentication 0 -OAuthAuthentication 0
-AdfsAuthentication 0
```

6. Configure Offline Address Book (OAB).

```
   C:\> Set-OabVirtualDirectory -WindowsAuthentication 0
-BasicAuthentication 1
-ExternalUrl https://mail.company.com/OAB
```

7. If Exchange 2013 Service Pack 1 (SP1) or later is installed, configure Basic authentication for the Messaging Application Programming Interface (MAPI) virtual directory. This feature is not available in the earlier versions of Exchange 2013.

```
   C:\> Set-MapiVirtualDirectory
-InternalURL http://mail.company.com/mapi
-ExternalURL https://mail.company.com/mapi
-IISAuthenticationMethods Basic
```

8. If Exchange 2013 SP 1 or later is installed, enable MAPI Over HTTP.

```
   C:\> Set-OrganizationConfig -MapiHTTPEnabled 1
```

Configuring Firewalls

If there is a firewall between the NGINX Plus server and other applications in your Exchange deployment, configure it to pass through traffic on the ports specified in the table. The columns represent the three types of applications that communicate with the NGINX Plus server – email clients, the NGINX Plus dashboard on your administrative network, and CASs – and the **x** indicates that the port must be open.

Port	Protocol	Email clients	Admin network	CASs
22	SSH		x	
25	SMTP	x		x
80	HTTP	x		
443	HTTPS	x		x
993	IMAPS	x		x
8080	HTTP (NGINX Plus dashboard)		x	

Installing NGINX Plus

You can install NGINX Plus on premise, in a private cloud, or in a public cloud such as the Amazon Web Service (AWS) Elastic Compute Cloud (EC2) or Microsoft Azure. See the instructions for your installation type:

- On premise or private cloud – Instructions for your Linux operating system at **NGINX repository**
- AWS EC2 – **Setting Up an NGINX Plus Environment on Amazon EC2**
- Microsoft Azure – **Setting up an NGINX Plus Environment on Microsoft Azure**

After installing NGINX Plus in any environment, you need to install an SSL certificate on the NGINX Plus server so that it can participate in encrypted communication with mail clients. Follow the instructions in the following section.

Configuring an SSL Certificate for NGINX Plus

Install an SSL certificate on the NGINX Plus server. There are several ways to obtain the required certificate, including the following. For your convenience, step-by-step instructions are provided for the second and third options.

- If you already have an SSL certificate for NGINX Plus installed on another system, copy it to the ***/etc/nginx/ssl/*** directory on the NGINX Plus server.
- If you need to request a new certificate from a certificate authority (CA) or your organization's security group, see "Creating a Certificate Request with the openssl Command" **below**.
- If you already have an SSL certificate for your Exchange servers, see "Exporting and Converting an SSL Certificate from an IIS Server" on **page 10**.

CREATING A CERTIFICATE REQUEST WITH THE OPENSSL COMMAND

1. Log in as the root user on a machine that has the **openssl** software installed.
2. Create a private key to be packaged in the certificate.

   ```
   root# openssl genrsa -out ~/company.com.key 2048
   ```

3. Create a backup of the key file and move it to a secure location. If you lose the key, the certificate becomes unusable.

   ```
   root# cp ~/company.com.key secure-dir/company.com.key.backup
   ```

4. Create a Certificate Signing Request (CSR) file.

   ```
   root# openssl req -new -sha256 -key ~/company.com.key \
         -out ~/company.com.csr
   ```

5. Request a certificate from a CA or your internal security group, providing the CSR file (**company.com.csr**). As a reminder, never share private keys (**.key** files) directly with third parties.

   The certificate needs to be Linux-compatible rather than Windows-compatible. If you request the certificate from a CA website yourself, choose NGINX or Apache (if available) when asked to select the server platform for which to generate the certificate.

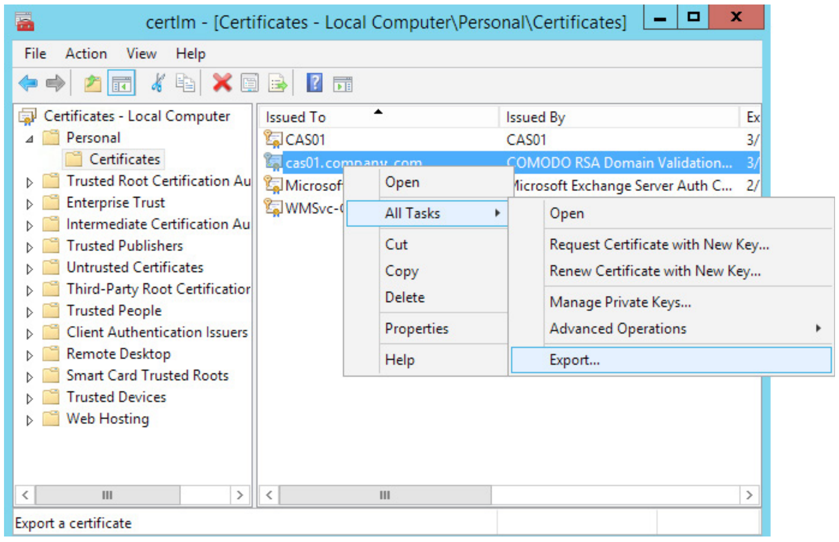
6. Copy or move the certificate file and associated key files to the ***/etc/nginx/ssl/*** directory on the NGINX Plus server.

EXPORTING AND CONVERTING AN SSL CERTIFICATE FROM AN IIS SERVER

On Windows systems, SSL certificates are packaged in a Public-Key Cryptography Standards (PKCS) archive file with extension **.pfx**. You need to export the **.pfx** file and convert the contents to the Linux-compatible format.

Working in the Microsoft Management Console, perform the following steps:

1. Open the **Certificates** snap-in.
2. In the navigation pane (directory tree), click the **Certificates** folder in the logical store for the certificate to export (in the following figure, it is **Personal > Certificates**).
3. In the main pane, right-click the certificate to be exported (in the following figure, it is **cas01.company.com**).
4. On the menu that pops up, select **All Tasks**, then click **Export.**



5. In the Certificate Export Wizard window that pops up, click **Yes, export the private key**. (This option appears only if the private key is marked as exportable and you have access to it.)

6. If prompted for a password (used to encrypt the private key before export), type it in the **Password** and **Confirm** fields. (Remember the password, as you need to provide it when importing the certificate to NGINX Plus.)

7. Click **Next**.

8. In **File name** field, type the filename and path to the location for storing the exported certificate and private key. Click **Next**, then **Finish**.

9. Copy the **.pfx** file to the NGINX Plus server.

Working on a machine that has the **openssl** software installed, perform the following steps:

10. Log in as the root user.

11. Create the private key file. You are prompted first for the password protecting the **.pfx** file (see Step 6 above), and then for a new password used to encrypt the private key file being created (**company.com.key.encrypted** in the sample command).

```
root# openssl pkcs12 -in exported-certs.pfx –nocerts \
      -out company.com.key.encrypted
```

12. Decrypt the key file. At the prompt, type the password you created in the previous step for the private key file.

```
root# openssl rsa -in company.com.key.encrypted \
      –out company.com.key
```

13. Create the certificate file.

```
root# openssl pkcs12 -in exported-cert.pfx -clcerts \
      -nokeys -out company.com.crt
```

14. Copy or move the certificate files and key files to the **/etc/nginx/ssl/** directory.

Configuring NGINX Plus for Exchange

You can configure NGINX Plus for either basic or enhanced load-balancing of Exchange traffic. Basic configuration provides complete load-balancing and reverse proxy functions. The enhanced configuration adds these features to make your deployment more reliable and easier to manage:

- Fine-grained URL location control – Exchange CASs interact with the various applications used by clients on different types of devices. Creating a separate `location` block for each application isolates the effect of an application outage to users of that application only. Other applications on the CAS continue to run normally.
- Health checks – Exchange includes a health-check mechanism for several applications that you can easily integrate with the health-check feature in NGINX Plus.
- Live activity monitoring – NGINX Plus includes a dashboard that provides key load and performance metrics in real time, including TCP metrics in NGINX Plus R6 and later.

For more information about these features, see "Completing the Configuration of Enhanced Load Balancing" on **page 22**.

Creating and Modifying Configuration Files

To reduce errors, this guide has you copy directives from files provided by NGINX, Inc. into your configuration files, instead of using a text editor to type in the directives yourself. Then you go through the sections in this guide (starting with “Configuring Global Settings” on **page 14**) to learn how to modify the directives as required for your deployment.

The provided files (one for basic load balancing and one for enhanced) have all directives together in a single file. If you are installing and configuring NGINX Plus on a fresh Linux system and using it only to load balance Exchange traffic, you can use the provided file as your main NGINX Plus configuration file, which by convention is **/etc/nginx/nginx.conf.**

We recommend, however, that instead of a single configuration file you use the scheme that is set up automatically when you install an NGINX Plus package (especially if you plan to expand your use of NGINX Plus to other purposes in future, or already have an existing NGINX Plus deployment). In the conventional scheme, the main configuration file is still called **/etc/nginx/nginx.conf**, but instead of listing all directives there you create separate configuration files for different functions and store the files in the **/etc/nginx/conf.d** directory. You then use the `include` directive in the main file to reference the separate files in the appropriate contexts.

To download the complete configuration file for basic or enhanced load balancing from the NGINX website, run the appropriate command:

```
root# curl http://nginx.com/resources/conf/exchange-basic.txt > \
      /etc/nginx/conf.d/exchange-basic.conf
```

or

```
root# curl http://nginx.com/resources/conf/exchange-enhanced.txt > \
      /etc/nginx/conf.d/exchange-enhanced.conf
```

(You can also access the URL in a browser and copy the text into the indicated file.)

To set up the conventional configuration scheme, perform these steps:

1. In the main **nginx.conf** file, add `http` and `stream` configuration blocks, if they do not already exist. (The standard placement is below any global directives; see "Configuring Global Settings" on **page 14**.) Add the indicated `include` directives (you can change the filenames if you wish).

```
http {
    include conf.d/exchange-http.conf;
}
stream {
    include conf.d/exchange-stream.conf;
}
```

You can also use wildcard notation to reference all files that pertain to a certain function or traffic type in the appropriate context block. For example, if you name all HTTP configuration files *function*-**http.conf** and all TCP configuration files *function*-**stream.conf**, these are appropriate include directives:

```
http {
    include conf.d/*-http.conf;
}
stream {
    include conf.d/*-stream.conf;
}
```

2. In the **/etc/nginx/conf.d** directory, create a new file for directives that pertain to Exchange HTTP and HTTPS traffic, giving it the name you chose in Step 1. Copy in the directives from the `http` configuration block in the complete configuration file. Remember not to copy the first line (`http {`) or the closing curly brace (`}`) for the block, because the `http` block you created in Step 1 already has them.

3. Create a new file for directives that pertain to Exchange TCP traffic, giving it the name you chose in Step 1. Copy in the directives from the `stream` configuration block in the complete configuration file. Again, do not copy the first line (`stream {`) or the closing curly brace (`}`).

For reference purposes, the full configuration files are also provided in this document:

- Full Configuration for Basic Load Balancing on **page 19**
- Full Configuration for Enhanced Load Balancing on **page 28**

However, we do not recommend that you copy from those sections directly. PDF does not use the same mechanisms for positioning text (such as line breaks and white space) as text editors do. In text copied from a PDF into an editor, lines run together and indenting of child statements in configuration blocks is missing or inconsistent. The absence of formatting does not present a problem for NGINX Plus, because (like many compilers) it ignores white space during parsing, relying solely on semicolons and curly braces as delimiters. The absence of white space does, however, make it more difficult for humans to interpret the configuration and modify it without making mistakes.

ABOUT RELOADING UPDATED CONFIGURATION

We recommend that each time you complete a set of updates to the configuration, you run the `nginx -t` command to test the configuration file for syntactic validity.

```
root# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

To tell NGINX Plus to start using the new configuration, run one of the following commands:

```
root# nginx -s reload
```

or

```
root# service nginx reload
```

Configuring Global Settings

Verify that the main **nginx.conf** file includes the following global directives, adding them as necessary.

```
user nginx;
worker_processes auto;

error_log /var/log/nginx/error.log info;

pid /var/run/nginx.pid;

events {

    worker_connections 1024;
}
# If using the standard configuration scheme, this is the usual
# location for 'http' and 'stream' blocks with 'include' directives.
```

Configuring TCP Load Balancing

Directives in the top-level `stream` configuration block configure TCP load balancing.

1. If placing all directives in the main **nginx.conf** file, copy in the complete `stream` configuration block from the file you downloaded in "Creating and Modifying Configuration Files" on page 12. The standard placement is below the global directives listed in "Configuring Global Settings" on page 14.

2. In the `upstream` block that defines the group of Internet Message Access Protocol (IMAP) servers across which NGINX Plus load balances traffic, include a `server` directive for each of your CASs.

```
# In the stream{} block
upstream exchange-imaps {

    zone exchange-imaps 64k;

    server 10.0.0.237:993; # Replace with IP address of a CAS

    server 10.0.0.238:993; # Replace with IP address of a CAS

}
```

3. In the `upstream` block that defines the group of Simple Mail Transfer Protocol (SMTP) servers across which NGINX Plus load balances traffic, include a `server` directive for each of your CASs.

```
# In the stream{} block
upstream exchange-smtp {
    zone exchange-smtp 64k;
    server 10.0.0.237:25; # Replace with IP address of a CAS
    server 10.0.0.238:25; # Replace with IP address of a CAS

}
```

4. The following server block defines the virtual server that proxies traffic on port 993 to the **exchange-imaps** upstream group configured in Step 2 .

```
# In the stream{} block
server {

    listen 993;
    status_zone exchange-imaps;

    proxy_pass exchange-imaps;
}
```

5. The following server block defines the virtual server that proxies traffic on port 25 to the **exchange-smtp** upstream group configured in Step 3. If you wish to change the port number from 25 (for example, to 587), change the `listen` directive.

```
# Add to the stream{} block
server {
    listen 25; # SMTP port can be changed here (to 587, for
               # example)
    status_zone exchange-smtp;
    proxy_pass exchange-smtp;
}
```

Configuring Global HTTP and HTTPS Settings

These directives in the top-level `http` configuration block configure global HTTP and HTTPS settings.

1. If placing all directives in the main **nginx.conf** file, copy in the complete `http` configuration block from the file you downloaded in “Creating and Modifying Configuration Files” on page 12. The standard placement is between the global directives listed in "Configuring Global Settings" on **page 14** and the `stream` configuration block.

2. These directives define the file in which access events are logged, and modify the default format of access log messages to include the `$upstream_addr` variable, which captures the address of the CAS.

```
# In the http{} block
access_log /var/log/nginx/access.log main;
log_format main '$remote_addr - $remote_user [$time_local] '
          '"$request" $status $body_bytes_sent '
          '"$http_user_agent" "$upstream_addr"';
```

3. These directives set the duration of the indicated timeouts to 3 hours each, to support HTTP long polling by Exchange clients.

```
# In the http{} block
keepalive_timeout 3h;
proxy_read_timeout 3h;
```

4. This directive enables use of the operating system's TCP_NODELAY option. (This option disables the aggregating of many small messages into a larger one, which is often done to reduce the number of packets being sent on the network.)

```
# In the http{} block
tcp_nodelay on;
```

5. This server block defines a virtual server that redirects HTTP traffic (on port 80) to HTTPS.

```
# In the http{} block
server {
    listen 80;
    location / {
        return 301 https://$host$request_uri;
    }
}
```

Configuring a Virtual Server for HTTPS Traffic

These directives define a virtual server for HTTPS traffic in a separate `server` block in the top-level `http` configuration block.

1. This `server` block defines the port for HTTPS traffic, activates SSL/TLS, and defines the server's status zone for live activity monitoring.

```
# In the http{} block
server {
    listen 443 ssl;
    status_zone exchange-combined;
}
```

2. This directive increases the default file upload size, which is required for Microsoft RPC Over HTTP. (Note that this directive does not set the maximum size of an email message.)

```
# In the server{} block for HTTPS traffic
client_max_body_size 2G;
```

3. These directives name the SSL certificate and private key, and disable use of any protocol less secure than TLS version 1.

```
# In the server{} block for HTTPS traffic
ssl_certificate /etc/nginx/ssl/company.com.crt;
ssl_certificate_key /etc/nginx/ssl/company.com.key;
ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
```

4. This `location` block redirects traffic from the main mail page (corresponding to /) to the Outlook Web App (OWA), which IIS does not do by default.

```
# In the server{} block for HTTPS traffic
location = / {
    return 301 "/owa/";
}
```

5. (Optional) If a browser requests the **favicon.ico** file and it is not available, this `location` block disables logging of any resulting errors and supplies an empty image file. Many applications provide a **favicon.ico** file containing the icon which appears on a page's tab in the browser to indicate the application. IIS by default does not make the **favicon.ico** file available for the main mail page.

```
# In the server{} block for HTTPS traffic
location = /favicon.ico {
  empty_gif;
  access_log off;
}
```

6. Proceed to the section for the final desired configuration:

- Completing the Configuration of Basic Load Balancing on page 18
- Completing the Configuration of Enhanced Load Balancing on page 22

Completing the Configuration of Basic Load Balancing

These directives complete the configuration for basic load balancing of Exchange traffic.

(To finalize enhanced load balancing instead, proceed to **page 22**.)

1. In the upstream block that defines the group of servers across which NGINX Plus load balances HTTPS traffic, include a `server` directive for each of your CASs.

```
# In the http{} block
upstream exchange {
    zone exchange-general 64k;
    server 10.0.0.237:443; # Replace with IP address of a CAS
    server 10.0.0.238:443; # Replace with IP address of a CAS
    sticky learn create=$remote_addr lookup=$remote_addr
           zone=client_sessions:10m timeout=3h;
}
```

2. In the `server` block for HTTPS traffic (described in "Configuring a Virtual Server for HTTPS Traffic" on **page 16**), this `location` block disables buffering of both uploads and downloads, as required by Microsoft RPC Over HTTP.

```
# In the server{} block for HTTPS traffic
location / {
    proxy_pass https://exchange;
    proxy_buffering off;
    proxy_http_version 1.1;
    proxy_request_buffering off;
    proxy_set_header Connection "Keep-Alive";
}
```

Full Configuration for Basic Load Balancing

Note: We do not recommend copy text from this section into your configuration files. See the alternative instructions in “Creating and Modifying Configuration Files” on page 12.

```
# For simplicity, all directives for basic load balancing of Microsoft
# Exchange traffic appear in this file. You can also create
# function-specific files in the /etc/nginx/conf.d directory, for example,
# separate files for the http{} and stream{} blocks in this file. Then use
# the ‘include’ directive in the main nginx.conf file to reference the
# function-specific files.

user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log info;
pid /var/run/nginx.pid;

events {
    worker_connections 1024;
}

http {
    access_log /var/log/nginx/access.log main;
    log_format main '$remote_addr - $remote_user [$time_local] '
              '"$request'' $status $body_bytes_sent '
              '"$http_user_agent" "$upstream_addr"';
    keepalive_timeout 3h;
    proxy_read_timeout 3h;
    tcp_nodelay on;

    upstream exchange {
        zone exchange-general 64k;
        server 10.0.0.237:443; # Replace with IP address of a CAS
        server 10.0.0.238:443; # Replace with IP address of a CAS
        sticky learn create=$remote_addr lookup=$remote_addr
               zone=client_sessions:10m timeout=3h;}
    }

    server {
        listen 80;
        location / {
            return 301 https://$host$request_uri;
        }
    }
    server {
        listen 443 ssl;
        client_max_body_size 2G;
        ssl_certificate /etc/nginx/ssl/company.com.crt;
        ssl_certificate_key /etc/nginx/ssl/company.com.key;
        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        status_zone exchange-combined;

       location = / {
           return 301 "/owa/";
       }

       location = /favicon.ico {
           empty_gif;
           access_log off;
       }

       location / {
           proxy_pass https://exchange;
           proxy_buffering off;
           proxy_http_version 1.1;
           proxy_request_buffering off;
           proxy_set_header Connection ''Keep-Alive'';
        }
    }
}

stream {
    upstream exchange-imaps {
        zone exchange-imaps 64k;
        server 10.0.0.237:993; # Replace with IP address of a CAS
        server 10.0.0.238:993; # Replace with IP address of a CAS
    }

    upstream exchange-smtp {
        zone exchange-smtp 64k;
        server 10.0.0.237:25; # Replace with IP address of a CAS
        server 10.0.0.238:25; # Replace with IP address of a CAS
    }

    server {
        listen 993;
        status_zone exchange-imaps;
        proxy_pass exchange-imaps;
    }
    server {
        listen 25; #SMTP port can be changed here (to 587, for example)
        status_zone exchange-smtp;
        proxy_pass exchange-smtp;
    }
}
```

Completing the Configuration of Enhanced Load Balancing

This section describes the configuration for enhanced load balancing of Exchange traffic. The enhancements improve the performance of your NGINX Plus deployment and make it easier to manage.

(To finalize basic load balancing instead, return to **page 18**.)

- Configuring Application-Specific Load Balancing **below**
- Enabling the Live Activity Monitoring Dashboard on **page 27**

Configuring Application-Specific Load Balancing

Exchange CASs interact with various applications used by clients on different types of devices. The clients access virtual directories and URIs specific to their application. To improve the performance of applications and of NGINX Plus, a separate `location` block for each application is configured for the features listed in the following table.

Application	**Virtual Directory or URI**	**Enhanced Features**
ActiveSync	**/Microsoft-Server-ActiveSync**	Health checks
Exchange Control Panel	**/ecp**	Restricted access and health checks
MAPI over HTTP	**/mapi**	Health checks
OWA	**/owa**	Health checks
RPC Over HTTP	**/rpc/rpcproxy.dll**	Unbuffered upload and download; session persistence

CONFIGURING GRANULAR URL LOCATION CONTROL

1. In the top-level `http` configuration block, separate `upstream` blocks for each application improve Exchange's overall reliability by isolating the effect of outages to just the affected applications. (In other words, creating separate upstream groups means that if an application or directory becomes unavailable, clients can still access the other applications and directories that are functioning and accessible.) In each `upstream` block, include a `server` directive for each of your CASs.

```
# In the http{} block
upstream exchange {
     zone exchange-general 64k;
     server 10.0.0.237:443;   # Replace with IP address of a CAS
     server 10.0.0.238:443;   # Replace with IP address of a CAS
}

upstream exchange-activesync {
     zone exchange-activesync 64k;
     server 10.0.0.237:443;   # Replace with IP address of a CAS
     server 10.0.0.238:443;   # Replace with IP address of a CAS
}
```

```
upstream exchange-ecp {
    zone exchange-ecp 64k;
    server 10.0.0.237:443;   # Replace with IP address of a CAS
    server 10.0.0.238:443;   # Replace with IP address of a CAS
}

upstream exchange-mapi {
    zone exchange-mapi 64k;
    server 10.0.0.237:443;   # Replace with IP address of a CAS
    server 10.0.0.238:443;   # Replace with IP address of a CAS
}

upstream exchange-owa {
    zone exchange-owa 64k;
    server 10.0.0.237:443;   # Replace with IP address of a CAS
    server 10.0.0.238:443;   # Replace with IP address of a CAS
}

upstream exchange-rpc {
    zone exchange-rpc 64k;
    server 10.0.0.237:443;   # Replace with IP address of a CAS
    server 10.0.0.238:443;   # Replace with IP address of a CAS
    sticky learn create=$remote_addr lookup=$remote_addr
           zone=client_sessions:10m timeout=3h;
}
```

2. In the `server` block for HTTP traffic (described in "Configuring a Virtual Server for HTTPS Traffic" on **page 16**), a separate `location` block for each client application configures different handling of each type of traffic:

- Clients that don't specify an application access the main page.

```
# In the server{} block for HTTPS traffic
location / {
    proxy_pass https://exchange;
}
```

- Administrators using the Exchange Control Panel (ECP) access **/ecp**. Presumably you want to restrict access to this location, and one of the simplest ways is to uncomment the following `allow` and `deny` directives, which allow access from your administrative network (substitute its IP address and prefix for **172.16.0.0/16**) and deny access to everyone else. You could also use other security methods, like SSL certificates or an additional layer of basic authentication.

```
# In the server{} block for HTTPS traffic
location /ecp {
    #allow 172.16.0.0/16; # Replace with your admin network
    #deny all;
    proxy_pass https://exchange-ecp;
}
```

- Outlook 2013 SP1 clients using MAPI Over HTTP access **/mapi.**

  ```
  # In the server{} block for HTTPS traffic
  location /mapi {
      proxy_pass https://exchange-mapi;
  }
  ```

- Mobile clients like iPhone and Android access the ActiveSync location (**/Microsoft-Server-ActiveSync**).

  ```
  # In the server{} block for HTTPS traffic
  location /Microsoft-Server-ActiveSync {
      proxy_pass https://exchange-activesync;
  }
  ```

- Clients using a browser for webmail access the OWA location (**/owa**).

  ```
  # In the server{} block for HTTPS traffic
  location /owa {
      proxy_pass https://exchange-owa;
  }
  ```

- Outlook Anywhere clients access the RPC Over HTTP location (**/rpc/rpcproxy.dll**). The directives disable buffering for both upload and download of content, as required by RPC Over HTTP.

  ```
  # In the server{} block for HTTPS traffic
  location /rpc/rpcproxy.dll {
      proxy_pass https://exchange-rpc;
      proxy_buffering off;
      proxy_http_version 1.1;
      proxy_request_buffering off;
      proxy_set_header Connection ''Keep-Alive'';
  }
  ```

CONFIGURING HEALTH CHECKS

For several applications, Exchange includes a health-check mechanism that you can easily integrate with the health-check feature in NGINX Plus. Specifically, you configure the NGINX Plus health check to succeed when the Exchange health check succeeds.

When an Exchange health check succeeds for an application, the following string is written at the end of the application's **healthcheck.htm** file.

```
200 OK<BR/>server-name.FQDN
```

Here's an example of the full contents of a **healthcheck.htm** file for the MAPI application:

```
root# curl -v https://mail.company.com/mapi/healthcheck.htm
> GET /mapi/healthcheck.htm HTTP/1.1
> User-Agent: curl/7.37.1
>   Host: mail.company.com
> Accept: */*
>
< HTTP/1.1 200 OK
* Server nginx/1.7.11 is not blacklisted
< Server: nginx/1.7.11
< Date: Thu, 02 Apr 2015 00:36:34 GMT
< Content-Length: 34
< Connection: keep-alive
< X-Powered-By: ASP.NET
< X-FEServer: CAS02
<
200 OK<BR/>CAS02.CORP.Company.com
```

These directives configure NGINX Plus health checks.

1. In the `server` block for HTTPS traffic (described in "Configuring a Virtual Server for HTTPS Traffic" on **page 16**), this `match` block checks for status code 200 and the string `200 OK` in the response body.

```
# In the server{} block for HTTPS traffic
match exchange-health {
    status 200;
    body ~ "200 OK";
}
```

2. In these `location` blocks (described in their basic form in "Configuring Granular URL Location Control" on **page 22**), additional `health_check` directives configure NGINX Plus health checks:

```
# In the server{} block for HTTPS traffic
location /ecp {
    #allow 172.16.0.0/16; # Replace with your admin network
    #deny all;
    proxy_pass https://exchange-ecp;
    health_check uri=/ecp/healthcheck.htm interval=3s
                 match=exchange-health;
}
```

```
location /mapi {
    proxy_pass https://exchange-mapi;
    health_check uri=/mapi/healthcheck.htm interval=3s
                 match=exchange-health;
}

location /owa {
    proxy_pass https://exchange-owa;
    health_check uri=/owa/healthcheck.htm interval=3s
                 match=exchange-health;
}
```

3. This `match` configuration block added to the `server` block for HTTPS traffic, along with this `health_check` directive added to the existing `location` block for Outlook Anywhere clients, directs RPC traffic away from servers that don't have Basic authentication enabled.

```
# In the server{} block for HTTPS traffic
match exchange-auth {
    status 401;
    header WWW-Authenticate ~ Basic;
}

location /rpc/rpcproxy.dll {
    proxy_pass https://exchange-rpc;
    proxy_buffering off;
    proxy_http_version 1.1;
    proxy_request_buffering off;
    proxy_set_header Connection ''Keep-Alive'';
    health_check uri=/rpc/rpcproxy.dll interval=3s;
                 match=exchange-auth
}
```

Enabling the Live Activity Monitoring Dashboard

NGINX Plus includes a live activity monitoring interface that provides key load and performance metrics in real time, including TCP metrics in NGINX Plus R6 and later. Statistics are reported through a RESTful JSON interface, making it very easy to feed the data to a custom or third-party monitoring tool. These instructions deploy the display dashboard that is built into NGINX Plus.

For detailed information about live activity monitoring, see the following documentation:

- **Live Activity Monitoring of NGINX Plus** (at nginx.com)
- **HTTP status module** (at nginx.org)

The quickest way to configure the built-in NGINX Plus dashboard is to download the sample configuration file from the NGINX, Inc. website:

```
root# curl http://nginx.com/resources/conf/status.txt > \
      /etc/nginx/conf.d/status.conf
```

If you are including all directives for Exchange load balancing in the main **nginx.conf** file, add this `include` directive to the top-level `http` configuration block in the main **nginx.conf** file:

```
# In the http{} block
include conf.d/status.conf;
```

If you are using the standard configuration scheme with function-specific files in the **/etc/nginx/conf.d** directory (see '"Creating and Modifying Configuration Files" on page 12), you can either add the `include` directive as shown above, or change the name of **status.conf** so it is captured by the wildcard you use in the existing `include` directive in the `http` block.

Comments in the **status.conf** file explain which directives you must customize for your deployment. In particular, the default settings in the sample configuration file allow anyone on any network to access the dashboard. We strongly recommend that you restrict access to the dashboard with one or more of the following methods:

- IP address-based access control lists (ACLs). In the sample configuration file, uncomment the `allow` and `deny` directives, and substitute the address of your administrative network for `10.0.0.0/8`. Only users on the specified network can access the status page.

  ```
  allow 10.0.0.0/8;
  deny all;
  ```

- HTTP basic authentication. In the sample configuration file, uncomment the `auth_basic` and `auth_basic_user_file` directives and add user entries to the **/etc/nginx/users** file (for example, by using an htpasswd generator).

  ```
  auth_basic on;
  auth_basic_user_file /etc/nginx/users;
  ```

- Client certificates, which are part of a complete configuration of SSL or TLS. For more information, see **NGINX SSL Termination** in the NGINX Plus Admin Guide and the documentation for the **HTTP SSL** module.

- Firewall. Configure your firewall to disallow outside access to the port for the dashboard **(8080** in the sample configuration file).

When you reload NGINX Plus with the new configuration, the NGINX Plus dashboard is available immediately at **http://*nginx-server-address*:8080.**

Full Configuration for Enhanced Load Balancing

Note: We do not recommend copy text from this section into your configuration files. See the alternative instructions in "Creating and Modifying Configuration Files" on page 12.

```
# For simplicity, all directives for enhanced load balancing of Microsoft
# Exchange traffic appear in this file. You can also create
# function-specific files in the /etc/nginx/conf.d directory, for example,
# separate files for the http{} and stream{} blocks in this file. Then use
# the 'include' directive in the main nginx.conf file to reference the
# function-specific files.

user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log debug;
pid /var/run/nginx.pid;

events {
    worker_connections 1024;
}

http {
    access_log /var/log/nginx/access.log main;
    log_format main '$remote_addr - $remote_user [$time_local] '
              '"$request" $status $body_bytes_sent '
              '"$http_user_agent" "$upstream_addr"';
    keepalive_timeout 3h;
    proxy_read_timeout 3h;
    tcp_nodelay on;

    # This 'include' directive is appropriate if you are placing all
    # Exchange-related directives in the main nginx.conf file. Adjust as
    # necessary if already using included feature-specific configuration
    # files.
    include conf.d/status.conf;

    upstream exchange {
        zone exchange-general 64k;
        server 10.0.0.237:443; # Replace with IP address of a CAS
        server 10.0.0.238:443; # Replace with IP address of a CAS
    }
```

```
 upstream exchange-activesync {
     zone exchange-activesync 64k;
     server 10.0.0.237:443; # Replace with IP address of a CAS
     server 10.0.0.238:443; # Replace with IP address of a CAS
 }


upstream exchange-ecp {
     zone exchange-ecp 64k;
     server 10.0.0.237:443; # Replace with IP address of a CAS
     server 10.0.0.238:443; # Replace with IP address of a CAS
 }

 upstream exchange-mapi {
     zone exchange-mapi 64k;
     server 10.0.0.237:443; # Replace with IP address of a CAS
     server 10.0.0.238:443; # Replace with IP address of a CAS
 }

 upstream exchange-owa {
     zone exchange-owa 64k;
     server 10.0.0.237:443; # Replace with IP address of a CAS
     server 10.0.0.238:443; # Replace with IP address of a CAS
 }

 upstream exchange-rpc {
     zone exchange-rpc 64k;
     server 10.0.0.237:443; # Replace with IP address of a CAS
     server 10.0.0.238:443; # Replace with IP address of a CAS
     sticky learn create=$remote_addr lookup=$remote_addr
            zone=client_sessions:10m timeout=3h;
 }

 match exchange-auth {
     status 401;
     header WWW-Authenticate ~ Basic;
 }

 match exchange-health {
     status 200;
     body ~ ''200 OK'';
 }

 server {
     listen 80;
     location / {
         return 301 https://$host$request_uri;
     }
 }
```

```
server {
    listen 443 ssl;
    client_max_body_size 2G;
    ssl_certificate /etc/nginx/ssl/company.com.crt;
    ssl_certificate_key /etc/nginx/ssl/company.com.key;
    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;

    location = / {
        return 301 "/owa/";
    }
    location = /favicon.ico {
        empty_gif;
        access_log off;
    }

    location / {
        proxy_pass https://exchange;
    }

    location /ecp {
        # Grant access to admin users only, by uncommenting the 'allow'
        # and 'deny' directives and substituting the IP address and
        # prefix of your admin network. Or configure more sophisticated
        # access control.
        #allow 172.16.0.0/16; # Replace with your admin network
        #deny all;
        proxy_pass https://exchange-ecp;
        health_check uri=/ecp/healthcheck.htm interval=3s
                     match=exchange-health;
    }

    location /mapi {
        proxy_pass https://exchange-mapi;
        health_check uri=/mapi/healthcheck.htm interval=3s
                     match=exchange-health;
    }

    location /Microsoft-Server-ActiveSync {
        proxy_pass https://exchange-activesync;
    }

    location /owa {
        proxy_pass https://exchange-owa;
        health_check uri=/owa/healthcheck.htm interval=3s
                     match=exchange-health;
    }
```

```
        location /rpc/rpcproxy.dll {
            proxy_http_version 1.1;
            proxy_set_header Connection ''Keep-Alive'';
            proxy_pass https://exchange-rpc;
            proxy_request_buffering off;
            proxy_buffering off;
            health_check uri=/rpc/rpcproxy.dll interval=3s
                         match=exchange-auth;
        }
    }
}

stream {
    upstream exchange-imaps {
        zone exchange-imaps 64k;
        server 10.0.0.237:993; # Replace with IP address of a CAS
        server 10.0.0.238:993; # Replace with IP address of a CAS
    }

    upstream exchange-smtp {
        zone exchange-smtp 64k;
        server 10.0.0.237:25; # Replace with IP address of a CAS
        server 10.0.0.238:25; # Replace with IP address of a CAS
    }

    server {
        listen 993;
        status_zone exchange-imaps;
        proxy_pass exchange-imaps;
    }

    server {
        listen 25; # SMTP port can be changed here (to 587, for example)
        status_zone exchange-smtp;
        proxy_pass exchange-smtp;
    }
}
```

For more information,

visit nginx.com, or

send us an email at

nginx-inquiries@nginx.com

NGINX