# Radiator® GBA/BSF Support module

Installation and reference manual for Radiator® GBA/ BSF Support module 1.9. Last revised on April 7, 2022

# Table of Contents

# 1. Introduction to Radiator GBA/BSF Support module

This document describes how to install and configure the Radiator GBA/BSF Support module.

For more information about GBA/BSF (General Bootstrapping Architecture/Bootstrapping Server Functionality) architecture, see Radiator GBA/BSF whitepaper [https://files.radiatorsoftware.com/radiator/whitepapers/GBA-BSF-whitepaper.pdf].

Radiator GBA/BSF Support module has 2 components, BSF (Bootstrapping Server Functionality) and NAF (Network Application Function)/AP (Application Proxy). The BSF component's hostname's format is usually `bsf.ims.mncXXX.mccYYY.pub.3gppnetwork.org` and it uses the following interfaces:

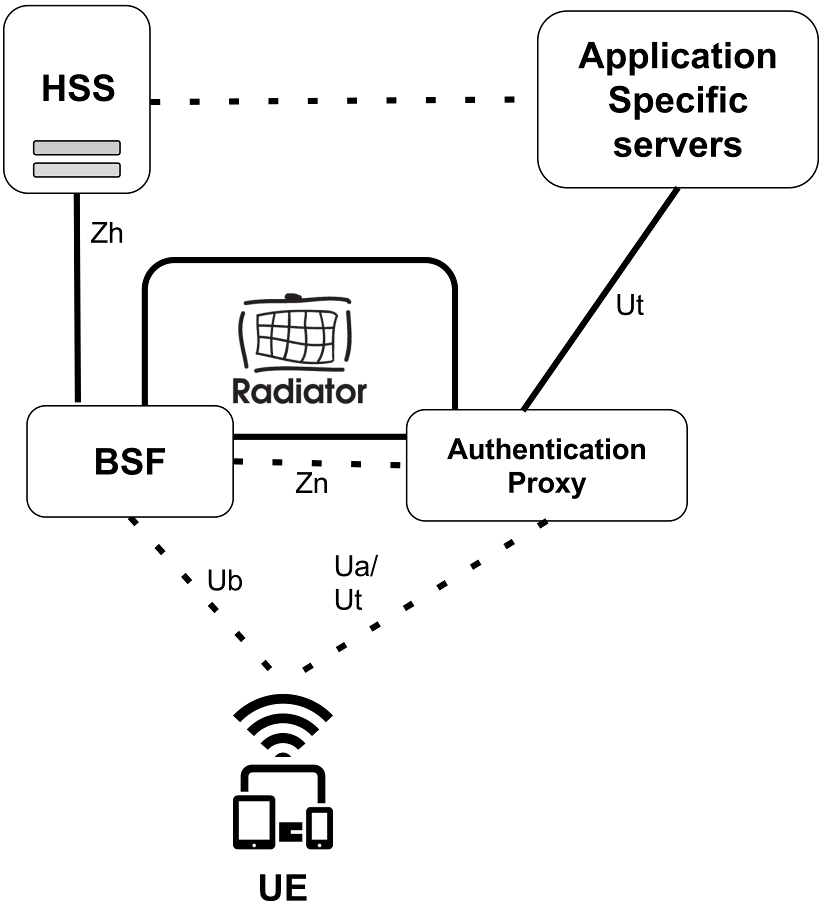
- HTTP/HTTPS Ub
- Diameter Zn
- Diameter Zh for connecting to the HSS (Home Subscriber Server) or DRA (Diameter Routing Agent)

The NAF/AP component's hostname's format is usually `xcap.ims.mncXXX.mccYYY.pub.3gppnetwork.org` and it uses the following interfaces:

- HTTP/HTTPS Ua and Ut
- Diameter Zn for connecting to the BSF

The following figure shows the basic architecture of GBA/BSF.

*Figure 1. GBA/BSF architecture*



# 2. Installing Radiator GBA/BSF Support module

This section provides instructions for installing Radiator GBA/BSF Support module. Radiator GBA/BSF Support module runs on a wide range of platforms and the installation procedure depends on the platform and the type of package selected.

Radiator GBA/BSF Support module can be configured to match various different environments. For example the BSF and AP configurations can be combined into a single configuration file if you are running BSF and AP components on the same host. See the installation instructions for more details.

## 2.1. Radiator GBA/BSF Support module prerequisites

You need the following software components for installing Radiator GBA/BSF Support module:

- Radiator version 4.25 or newer. For more information, see Radiator website [https://radiatorsoftware.com/products/radiator/].
- Radiator Carrier Module version 1.6 or newer. For more information, see Radiator Carrier website [https://radiatorsoftware.com/products/radiator-carrier-pack/].
- Perl version 5.8.8 or newer.
- The following Perl libraries:
  - `XML::LibXML`
  - `Digest::SHA`
  - `Digest::MD5`
  - `MIME::Base64`
  - `Cache::FastMmap`
  - `Storable`
- OpenResty application server with NGINX, LuaJIT, and NGINX add-on modules version 1.13.0 or newer. For more information, see OpenResty website [https://openresty.org/] and OpenResty Linux Packages [https://openresty.org/en/linux-packages.html].
  - OpenResty OpenSSL library required for TLS.

## 2.2. Installing Radiator GBA/BSF Support module from RPM repository

These packages have been tested on Red Hat Enterprise Linux 7 and newer, CentOS 7 and CentOS 8, and Oracle Linux 7 and newer.

To install Radiator GBA/BSF Support module from RPM repository:

1. Install all prerequisites: Radiator `el7.noarch.rpm` or `el8.noarch.rpm`, Radiator Carrier Module `el7.noarch.rpm` or `el8.noarch.rpm`, Perl and listed modules, and OpenResty as described in Radiator GBA/BSF Support module prerequisites on page 2
   - Radiator GBA/BSF Support module can pull automatically Radiator, Radiator Carrier Module and OpenResty from their respective repositories if those repositories have been configured on the system. For Radiator repositories see Radiator Linux package repositories website [https://downloads.radiatorsoftware.com/repo/] and for OpenResty repositories OpenResty Linux Packages website. [https://openresty.org/en/linux-packages.html]

- Radiator GBA/BSF Support module's testing repository contains `perl-Cache-FastMmap-1.48-1` RPM for convenience. If the testing repository has been enabled, Radiator GBA/BSF Support module can pull `perl-Cache-FastMmap-1.48-1` automatically.

2. Follow the instructions on Radiator Linux package repositories website [https://downloads.radiatorsoftware.com/repo/] *Radiator GBA/BSF pack* or *Radiator GBA/BSF pack evaluation* section on how to enable the repository and install Radiator GBA/BSF Support module from it.

3. Copy the example `radiator-xxx.conf` configuration matching your environment from `/opt/radiator/radiator-gba-bsf/goodies/` directory into the `/etc/radiator/` directory as `radiator.conf`. For example if you are running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `radiator-bsf-ap-combined.conf`.
   Example:

   ```
   sudo cp /opt/radiator/radiator-gba-bsf/goodies/radiator-bsf.conf /etc/radiator/radiator.conf
   ```

4. Copy the example `nginx-xxx.conf` configuration matching your environment from `/opt/radiator/radiator-gba-bsf/goodies/` directory into `/etc/nginx/` directory as `nginx.conf`. For example if you are running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `nginx-bsf-ap-combined.conf`.
   Example:

   ```
   sudo cp /opt/radiator/radiator-gba-bsf/goodies/nginx-bsf.conf /etc/nginx/nginx.conf
   ```

5. Edit the example configuration files to match your environment. Use Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] and Radiator GBA/BSF reference manual [https://files.radiatorsoftware.com/gba-bsf/radiator-gba-bsf-ref.pdf] for checking different configuration options and values.
   Check especially the following configuration options for Radiator configuration:

   - *Trace 4* (debug) level logging is useful during testing, but for production lower level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within configuration clause **`<IntegratorGBA BSF>`**, change *HSSServerName* to DRA’s Diameter Host name if you are connecting Radiator to DRA instead of HSS.
   - Within the configuration clause **`<DiaPeerDef Origin-Host=hss.mnc001.mcc001.3gppnetwork.org>`**, change *Peer* IP address to match your HSS or DRA.
   - If you want to enable Diameter statistics, uncomment the configuration clause **`<DiaStatsLog FILE>`**.

   For OpenResty/NGINX configuration:

   - For all **`error_log`** instances, *debug* log level is good for testing but for production *error* log level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within **`upstream UPSTREAM_SERVERS`** block, change and add IP address or addresses of MMTel/MTAS.
   - Search for **`HTTP/HTTPS server for AP (authenticating proxy)`** and within the server block, change **`proxy_pass`** directive to *proxy_pass http://UPSTREAM_SERVERS;* or *proxy_pass http://UPSTREAM_SERVERS/Your_AS_Custom_Prefix* if your MMTel/MTAS expects a custom prefix for XCAP requests.

6. Start the OpenResty/NGINX web server.

   ```
   sudo systemctl start openresty
   ```

7. Start the Radiator server.

   ```
   sudo systemctl start radiator
   ```

8. To ensure your system is working correctly, check the log files. They are located in the following directories by default:

   - `/var/log/radiator/`
     Example:

     ```
     tail -n 100 /var/log/radiator/radiator.log
     ```

   - `/var/log/nginx/`

     - When testing GBA authentication with an UE, Radiator and NGINX log files will show how BSF bootstrapping and AP authentication proceed.
       Example:

       ```
       tail -n 100 /var/log/nginx/error.log
       ```

       ```
       tail -n 100 /var/log/nginx/bsf_error.log
       ```

       ```
       tail -n 100 /var/log/nginx/ap_error.log
       ```

     - NGINX web server access log files will show all HTTP requests, one per line.
       Example:

       ```
       tail -n 100 /var/log/nginx/bsf_access.log
       ```

       ```
       tail -n 100 /var/log/nginx/ap_access.log
       ```

   - See that UNIX domain socket files have been created. OpenResty/NGINX uses UNIX domain socket files `/var/run/radiator/sock/radiator-bsf.sock` and `/var/run/radiator/sock/radiator-naf.sock` to communicate with Radiator.

## 2.3. Installing Radiator GBA/BSF Support module from deb repository

These packages have been tested on Ubuntu 16.04 and newer, and Debian 9 and newer.

To install Radiator GBA/BSF Support module from deb repository:

1. Install all prerequisites: Radiator `all.deb`, Radiator Carrier Module `all.deb`, Perl and listed modules, and OpenResty as described in Radiator GBA/BSF Support module prerequisites on page 2

   - Radiator GBA/BSF Support module can pull automatically Radiator, Radiator Carrier Module and OpenResty from their respective repositories if those repositories have been configured on the system. For Radiator repositories see Radiator Linux package repositories website [https://downloads.radiatorsoftware.com/repo/] and for OpenResty repositories OpenResty Linux Packages website [https://openresty.org/en/linux-packages.html]

2. Follow the instructions on Radiator Linux package repositories website [https://downloads.radiatorsoftware.com/repo/] *Radiator GBA/BSF pack* or *Radiator GBA/BSF pack evaluation* section on how to enable the repository and install Radiator GBA/BSF Support module from it.

3. Copy the example `radiator-xxx.conf` configuration matching your environment from `/opt/radiator/radiator-gba-bsf/goodies/` directory into the `/etc/radiator/` directory as `radiator.conf`. For example if you are running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `radiator-bsf-ap-combined.conf`.
   Example:

   ```
   sudo cp /opt/radiator/radiator-gba-bsf/goodies/radiator-bsf.conf /etc/radiator/radiator.conf
   ```

4. Copy the example `nginx-xxx.conf` configuration matching your environment from `/opt/radiator/radiator-gba-bsf/goodies/` directory into `/etc/nginx/` directory as `nginx.conf`. For example if you are running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `nginx-bsf-ap-combined.conf`.
   Example:

   ```
   sudo cp /opt/radiator/radiator-gba-bsf/goodies/nginx-bsf.conf /etc/nginx/nginx.conf
   ```

5. Edit the example configuration files to match your environment. Use Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] and Radiator GBA/BSF reference manual [https://files.radiatorsoftware.com/gba-bsf/radiator-gba-bsf-ref.pdf] for checking different configuration options and values.
   Check especially the following configuration options for Radiator configuration:

   - *Trace 4* (debug) level logging is useful during testing, but for production lower level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within configuration clause **`<IntegratorGBA BSF>`**, change *HSSServerName* to DRA’s Diameter Host name if you are connecting Radiator to DRA instead of HSS.
   - Within the configuration clause **`<DiaPeerDef Origin-Host=hss.mnc001.mcc001.3gppnetwork.org>`**, change *Peer* IP address to match your HSS or DRA.
   - If you want to enable Diameter statistics, uncomment the configuration clause **`<DiaStatsLog FILE>`**.

   For OpenResty/NGINX configuration:

   - For all **`error_log`** instances, *debug* log level is good for testing but for production *error* log level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within **`upstream UPSTREAM_SERVERS`** block, change and add IP address or addresses of MMTel/MTAS.
   - Search for **`HTTP/HTTPS server for AP (authenticating proxy)`** and within the server block, change **`proxy_pass`** directive to *proxy_pass http://UPSTREAM_SERVERS;* or *proxy_pass http://UPSTREAM_SERVERS/Your_AS_Custom_Prefix* if your MMTel/MTAS expects a custom prefix for XCAP requests.

6. Start the OpenResty/NGINX web server.

   ```
   sudo systemctl start openresty
   ```

7. Start the Radiator server.

   ```
   sudo systemctl start radiator
   ```

8. To ensure your system is working correctly, check the log files. They are located in the following directories by default:

   - `/var/log/radiator/`
     Example:

     ```
     tail -n 100 /var/log/radiator/radiator.log
     ```

   - `/var/log/nginx/`

     - When testing GBA authentication with an UE, Radiator and NGINX log files will show how BSF bootstrapping and AP authentication proceed.
       Example:

       ```
       tail -n 100 /var/log/nginx/error.log
       ```

       ```
       tail -n 100 /var/log/nginx/bsf_error.log
       ```

       ```
       tail -n 100 /var/log/nginx/ap_error.log
       ```

     - NGINX web server access log files will show all HTTP requests, one per line.
       Example:

       ```
       tail -n 100 /var/log/nginx/bsf_access.log
       ```

       ```
       tail -n 100 /var/log/nginx/ap_access.log
       ```

   - See that UNIX domain socket files have been created. OpenResty/NGINX uses UNIX domain socket files `/var/run/radiator/sock/radiator-bsf.sock` and `/var/run/radiator/sock/radiator-naf.sock` to communicate with Radiator.

## 2.4. Installing Radiator GBA/BSF Support module from RPM package

These packages have been tested on Red Hat Enterprise Linux 7 and newer, CentOS 7 and CentOS 8, and Oracle Linux 7 and newer.

To install Radiator GBA/BSF Support module from RPM package:

1. Install all prerequisites: Radiator `el7.noarch.rpm` or `el8.noarch.rpm`, Radiator Carrier Module `el7.noarch.rpm` or `el8.noarch.rpm`, Perl and listed modules, and OpenResty as described in Radiator GBA/BSF Support module prerequisites on page 2

2. Download the Radiator GBA/BSF Support module `el7.noarch.rpm` or `el8.noarch.rpm` distribution from Radiator GBA/BSF Support module download website [https://downloads.radiatorsoftware.com/packages/gba-bsf/downloads/].

3. Install the package.

   ```
   sudo yum install radiator-gba-bsf-1.8-nn.el7.noarch.rpm
   ```

   or

   ```
   sudo yum install radiator-gba-bsf-1.8-nn.el8.noarch.rpm
   ```

4. Copy the example `radiator-xxx.conf` configuration matching your environment from `/opt/radiator/radiator-gba-bsf/goodies/` directory into the `/etc/radiator/` directory as `radiator.conf`. For example if you are running BSF and AP components on the same host, combine

the BSF and AP configurations into a single configuration file as shown in the example configuration file `radiator-bsf-ap-combined.conf`.
Example:

```
sudo cp /opt/radiator/radiator-gba-bsf/goodies/radiator-bsf.conf /etc/radiator/radiator.conf
```

5. Copy the example `nginx-xxx.conf` configuration matching your environment from `/opt/radiator/radiator-gba-bsf/goodies/` directory into `/etc/nginx/` directory as `nginx.conf`. For example if you are running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `nginx-bsf-ap-combined.conf`.
   Example:

   ```
   sudo cp /opt/radiator/radiator-gba-bsf/goodies/nginx-bsf.conf /etc/nginx/nginx.conf
   ```

6. Edit the example configuration files to match your environment. Use Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] and Radiator GBA/BSF reference manual [https://files.radiatorsoftware.com/gba-bsf/radiator-gba-bsf-ref.pdf] for checking different configuration options and values.
   Check especially the following configuration options for Radiator configuration:

   - *Trace 4* (debug) level logging is useful during testing, but for production lower level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within configuration clause **`<IntegratorGBA BSF>`**, change *HSSServerName* to DRA’s Diameter Host name if you are connecting Radiator to DRA instead of HSS.
   - Within the configuration clause **`<DiaPeerDef Origin-Host=hss.mnc001.mcc001.3gppnetwork.org>`**, change *Peer* IP address to match your HSS or DRA.
   - If you want to enable Diameter statistics, uncomment the configuration clause **`<DiaStatsLog FILE>`**.

   For OpenResty/NGINX configuration:

   - For all **`error_log`** instances, *debug* log level is good for testing but for production *error* log level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within **`upstream UPSTREAM_SERVERS`** block, change and add IP address or addresses of MMTel/MTAS.
   - Search for **`HTTP/HTTPS server for AP (authenticating proxy)`** and within the server block, change **`proxy_pass`** directive to *proxy_pass http://UPSTREAM_SERVERS;* or *proxy_pass http://UPSTREAM_SERVERS/Your_AS_Custom_Prefix* if your MMTel/MTAS expects a custom prefix for XCAP requests.

7. Start the OpenResty/NGINX web server.

   ```
   sudo systemctl start openresty
   ```

8. Start the Radiator server.

   ```
   sudo systemctl start radiator
   ```

9. To ensure your system is working correctly, check the log files. They are located in the following directories by default:

- `/var/log/radiator/`
  Example:

  ```
  tail -n 100 /var/log/radiator/radiator.log
  ```

- `/var/log/nginx/`
  - When testing GBA authentication with an UE, Radiator and NGINX log files will show how BSF bootstrapping and AP authentication proceed.
    Example:

    ```
    tail -n 100 /var/log/nginx/error.log
    ```

    ```
    tail -n 100 /var/log/nginx/bsf_error.log
    ```

    ```
    tail -n 100 /var/log/nginx/ap_error.log
    ```

  - NGINX web server access log files will show all HTTP requests, one per line.
    Example:

    ```
    tail -n 100 /var/log/nginx/bsf_access.log
    ```

    ```
    tail -n 100 /var/log/nginx/ap_access.log
    ```

- See that UNIX domain socket files have been created. OpenResty/NGINX uses UNIX domain socket files `/var/run/radiator/sock/radiator-bsf.sock` and `/var/run/radiator/sock/radiator-naf.sock` to communicate with Radiator.

## 2.5. Installing Radiator GBA/BSF Support module from deb package

These packages have been tested on Ubuntu 16.04 and newer, and Debian 9 and newer.

To install Radiator GBA/BSF Support module from deb package:

1. Install all prerequisites: Radiator `all.deb`, Radiator Carrier Module `all.deb`, Perl and listed modules, and OpenResty as described in Radiator GBA/BSF Support module prerequisites on page 2

2. Download the Radiator GBA/BSF Support module `all.deb` distribution from Radiator GBA/BSF Support module download website [https://downloads.radiatorsoftware.com/packages/gba-bsf/downloads/].

3. Install the package.

   ```
   sudo apt install radiator-gba-bsf_1.8-nn_all.deb
   ```

4. Copy the example `radiator-xxx.conf` configuration matching your environment from `/opt/radiator/radiator-gba-bsf/goodies/` directory into the `/etc/radiator/` directory as `radiator.conf`. For example if you are running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `radiator-bsf-ap-combined.conf`.
   Example:

   ```
   sudo cp /opt/radiator/radiator-gba-bsf/goodies/radiator-bsf.conf /etc/radiator/radiator.conf
   ```

5. Copy the example `nginx-xxx.conf` configuration matching your environment from `/opt/radiator/radiator-gba-bsf/goodies/` directory into `/etc/nginx/` directory as `nginx.conf`. For example if you are running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `nginx-bsf-ap-combined.conf`.

Example:

```
sudo cp /opt/radiator/radiator-gba-bsf/goodies/nginx-bsf.conf /etc/nginx/nginx.conf
```

6. Edit the example configuration files to match your environment. Use Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] and Radiator GBA/BSF reference manual [https://files.radiatorsoftware.com/gba-bsf/radiator-gba-bsf-ref.pdf] for checking different configuration options and values.
   Check especially the following configuration options for Radiator configuration:

   - *Trace 4* (debug) level logging is useful during testing, but for production lower level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within configuration clause **`<IntegratorGBA BSF>`**, change *HSSServerName* to DRA's Diameter Host name if you are connecting Radiator to DRA instead of HSS.
   - Within the configuration clause **`<DiaPeerDef Origin-Host=hss.mnc001.mcc001.3gppnetwork.org>`**, change *Peer* IP address to match your HSS or DRA.
   - If you want to enable Diameter statistics, uncomment the configuration clause **`<DiaStatsLog FILE>`**.

   For OpenResty/NGINX configuration:

   - For all **`error_log`** instances, *debug* log level is good for testing but for production *error* log level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within **`upstream UPSTREAM_SERVERS`** block, change and add IP address or addresses of MMTel/MTAS.
   - Search for **`HTTP/HTTPS server for AP (authenticating proxy)`** and within the server block, change **`proxy_pass`** directive to *proxy_pass http://UPSTREAM_SERVERS;* or *proxy_pass http://UPSTREAM_SERVERS/Your_AS_Custom_Prefix* if your MMTel/MTAS expects a custom prefix for XCAP requests.

7. Start the OpenResty/NGINX web server.

   ```
   sudo systemctl start openresty
   ```

8. Start the Radiator server.

   ```
   sudo systemctl start radiator
   ```

9. To ensure your system is working correctly, check the log files. They are located in the following directories by default:

   - `/var/log/radiator/`
     Example:

     ```
     tail -n 100 /var/log/radiator/radiator.log
     ```

   - `/var/log/nginx/`
     - When testing GBA authentication with an UE, Radiator and NGINX log files will show how BSF bootstrapping and AP authentication proceed.
       Example:

```
tail -n 100 /var/log/nginx/error.log
```

```
tail -n 100 /var/log/nginx/bsf_error.log
```

```
tail -n 100 /var/log/nginx/ap_error.log
```

  – NGINX web server access log files will show all HTTP requests, one per line.
    Example:

```
tail -n 100 /var/log/nginx/bsf_access.log
```

```
tail -n 100 /var/log/nginx/ap_access.log
```

- See that UNIX domain socket files have been created. OpenResty/NGINX uses UNIX domain socket files `/var/run/radiator/sock/radiator-bsf.sock` and `/var/run/radiator/sock/radiator-naf.sock` to communicate with Radiator.

## 2.6. Installing Radiator GBA/BSF Support module from source package

RPM and deb packages are the recommended installation methods. Installing Radiator GBA/BSF Support module from a source package should only be used when specifically instructed by Radiator Software.

To install Radiator GBA/BSF Support module from source package:

1. Install all prerequisites: Radiator, Radiator Carrier Module, Perl and listed modules, and OpenResty as described in Radiator GBA/BSF Support module prerequisites on page 2

2. Download the Radiator GBA/BSF Support module source distribution `Radius-GBA-BSF-1.8-nn.tgz` from Radiator GBA/BSF Support module download website [https://downloads.radiatorsoftware.com/packages/gba-bsf/downloads/] and unpack it.

3. Change to `Radius-GBA-BSF-1.8-nn` directory and prepare for installation.

```
cd Radius-GBA-BSF-1.8-nn
```

```
perl Makefile.PL
```

4. Run the installation. You may need the root access rights for running this command.

```
make install
```

   Instead of steps 3 and 4, you can copy the Radiator GBA/BSF Support module distribution's `Radius/` directory into the `/opt/radiator/radiator/Radius/` directory and copy `lua/` directory into the `/opt/radiator/radiator-gba-bsf/` directory.

5. Copy the example `radiator-xxx.conf` configuration matching your environment from `Radius-GBA-BSF-1.8-nn/goodies/` directory into the `/etc/radiator/` directory as `radiator.conf`.
   For example if you are running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `radiator-bsf-ap-combined.conf`.
   Example:

```
sudo cp Radius-GBA-BSF-1.8-nn/goodies/radiator-bsf.conf /etc/radiator/radiator.conf
```

6. Copy the example `nginx-xxx.conf` configuration matching your environment from `Radius-GBA-BSF-1.8-nn/goodies/` directory into `/etc/nginx/` directory as `nginx.conf`. For example if you are

running BSF and AP components on the same host, combine the BSF and AP configurations into a single configuration file as shown in the example configuration file `nginx-bsf-ap-combined.conf`.
Example:

```
sudo cp Radius-GBA-BSF-1.8-nn/goodies/nginx-bsf.conf /etc/nginx/nginx.conf
```

7. Edit the example configuration files to match your environment. Use Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] and Radiator GBA/BSF reference manual [https://files.radiatorsoftware.com/gba-bsf/radiator-gba-bsf-ref.pdf] for checking different configuration options and values.
   Check especially the following configuration options for Radiator configuration:

   - *Trace 4* (debug) level logging is useful during testing, but for production lower level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within configuration clause **`<IntegratorGBA BSF>`**, change *HSSServerName* to DRA's Diameter Host name if you are connecting Radiator to DRA instead of HSS.
   - Within the configuration clause **`<DiaPeerDef Origin-Host=hss.mnc001.mcc001.3gppnetwork.org>`**, change *Peer* IP address to match your HSS or DRA.
   - If you want to enable Diameter statistics, uncomment the configuration clause **`<DiaStatsLog FILE>`**.

   For OpenResty/NGINX configuration:

   - For all **`error_log`** instances, *debug* log level is good for testing but for production *error* log level is recommended.
   - Change all *mnc001.mcc001* parts within the configuration to match your MNC and MCC codes.
   - Within **`upstream UPSTREAM_SERVERS`** block, change and add IP address or addresses of MMTel/MTAS.
   - Search for **`HTTP/HTTPS server for AP (authenticating proxy)`** and within the server block, change **`proxy_pass`** directive to *proxy_pass http://UPSTREAM_SERVERS;* or *proxy_pass http://UPSTREAM_SERVERS/Your_AS_Custom_Prefix* if your MMTel/MTAS expects a custom prefix for XCAP requests.

8. Start the OpenResty/NGINX web server.

   ```
   sudo systemctl start openresty
   ```

9. Start the Radiator server.

   ```
   sudo systemctl start radiator
   ```

10. To ensure your system is working correctly, check the log files. They are located in the following directories by default:

    - `/var/log/radiator/`
      Example:

      ```
      tail -n 100 /var/log/radiator/radiator.log
      ```

    - `/var/log/nginx/`
      - When testing GBA authentication with an UE, Radiator and NGINX log files will show how BSF bootstrapping and AP authentication proceed.

Example:

```
tail -n 100 /var/log/nginx/error.log
```

```
tail -n 100 /var/log/nginx/bsf_error.log
```

```
tail -n 100 /var/log/nginx/ap_error.log
```

– NGINX web server access log files will show all HTTP requests, one per line.
Example:

```
tail -n 100 /var/log/nginx/bsf_access.log
```

```
tail -n 100 /var/log/nginx/ap_access.log
```

- See that UNIX domain socket files have been created. OpenResty/NGINX uses UNIX domain socket files `/var/run/radiator/sock/radiator-bsf.sock` and `/var/run/radiator/sock/radiator-naf.sock` to communicate with Radiator.

# 3. Configuring Radiator GBA/BSF Support module

This section describes the configurable parameters of Radiator GBA/BSF Support module.

See `goodies/radiator-bsf-ap-combined.conf` for a configuration sample.

## 3.1. <AuthBy DiaBSF>

This section describes the configuration parameters of *<AuthBy DiaBSF>*. Apart from the parameters listed here, *<AuthBy DiaBSF>* inherits other parameters from *AuthGeneric*. These parameters are documented in Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <AuthBy xxxxxx>.

### 3.1.1. BSFZn

This object list lists the used BSFZn clauses.

### Example

```
BSFZn BSF-Zn
```

## 3.2. <BSFZn>

This section describes the configuration parameters of *<BSFZn>*.

### 3.2.1. Identifier

This is an optional parameter, which defines the name of the specific *<BSFZn>* clause and its configuration.

### Example

```
Identifier BSF-Zn
```

### 3.2.2. CacheFile

Location of shared memory cache mmap file. Defaults to `/tmp/sharefile-$pid-$time-$random`. Special variable *%0* will contain the suffix `-$pid-$time-$random`.

### 3.2.3. CacheExpiry

This string defines the expiration time of entries in shared memory cache. The value **0** means there is no explicit expiration time, the least recently used value is expired first when needed. This value can be expressed in minutes (**m**), hours (**h**), or days (**d**). The default value is **1d**.

### 3.2.4. CacheSize

This string defines the size of a shared memory cache. This value can be expressed in kilobytes (**k**) or megabytes (**m**). The default value is **100m**.

### 3.2.5. OriginHost

This string defines the name that *<BSFZn>* uses to identify itself to the Diameter peers. It is sent to the Diameter peers in the Diameter BIR (Bootstrapping Info Request) and BIA (Bootstrapping Info Answer) messages. The Diameter peers use *OriginHost* to determine whether they have connected to the correct peer. *OriginHost* must be specified.

#### Example

```
OriginHost bsf.ims.mnc001.mcc001.pub.3gppnetwork.org
```

### 3.2.6. OriginRealm

This string defines the name of the realm the *<BSFZn>* uses. It is sent to the Diameter peers in the BIR and BIA messages. The peer uses *OriginRealm* to determine which requests are routed to this Radiator instance. *OriginRealm* must be specified.

#### Example

```
OriginRealm ims.mnc001.mcc001.pub.3gppnetwork.org
```

### 3.2.7. RequireUSSForNaf

This flag defines whether the USS (User Security Settings) for NAF is required to exist for the subscriber. This is not set by default.

### 3.2.8. SendIMPI

This flag defines whether the subscriber's IMPI (IP Multimedia Private Identity) is sent back to NAF within Zn BIA. This is not set by default.

## 3.3. <DiaPeerDef>

This section describes the configuration parameters for *<DiaPeerDef>*. *<DiaPeerDef>* defines the Diameter peer this Radiator instance connects to. Both Radiator instance and the Diameter peer can initiate the connection.

A minimal Radiator GBA/BSF Support module configuration requires one *<DiaPeerDef>* clause for all used Diameter-based AuthBys. If there is no *<ServerDIAMETERTelco>* clause defined, *<DiaPeerDef>* clauses must have the *Initiator* flag set to connect to the Diameter peers.

A *<ServerDIAMETERTelco>* clause allows accepting incoming Diameter connections. When the *<ServerDIAMETERTelco>* is configured, Radiator acts as a Diameter responser. The settings for the connecting peers are fetched from the *<DiaPeerDef>* clauses.The clauses are matched against the incoming CER (Capabilities Exchange Request) from the peer.

**Note**

At least one `<DiaPeerDef>` clause is always required.

If the `<ServerDIAMETERTelco>` clause is configured but there are no `<DiaPeerDef>` clauses, the incoming CER messages are rejected by Radiator. A `<DiaPeerDef>` is required to form a successful CEA (Capabilities Exchange Answer) back to the peer.

**Note**

A `<DiaPeerDef>` with an empty parameter list matches to any Diameter peer. This is useful when defining default settings for incoming connections from any Diameter peer.

## 3.3.1. Identifier

This is an optional parameter, which defines the name of the specific `<DiaPeerDef>` clause and its configuration.

## 3.3.2. AddToRequestFromDia

This parameter defines the Diameter attributes, which are added to a request object in addition with *`OriginHost`* on page 14 and *`OriginRealm`* on page 15. The request object is created when a Diameter request message is received. The request object is then sent to the handler with the correct application AuthBy for this request.

The request object contains reference to the incoming Diameter request. The chosen Diameter application adds the reference to the Diameter answer.

## 3.3.3. PreHandlerHook

This is an optional parameter, which defines the Perl function that is called before the request object is sent to the handlers. The only passed argument is the reference to the current request object.

## 3.3.4. NoReplyHook

This is an optional parameter, which defines the Perl function that is called if no reply is received from any Diameter peer.

## 3.3.5. NoreplyTimeout

This integer defines how soon, in seconds, *`NoReplyHook`* on page 14 is called if the request stored in proxy does not receive a reply. The default value is **`5`**.

## 3.3.6. ProductName

This is an optional parameter, which defines the name of the specific Diameter peer. If defined, it is sent to the other Diameter peers within the CER and CEA messages. The default value is **`Radiator`**.

## 3.3.7. OriginHost

This string defines the name that `<ServerDIAMETERTelco>` uses to identify itself to the Diameter peers. It is sent to the Diameter peers in the Diameter CER and CEA messages. The Diameter peers use *`OriginHost`* to determine whether they have connected to the correct peer. *`OriginHost`* must be specified.

### 3.3.8. OriginRealm

This string defines the name of the Realm the *<ServerDIAMETERTelco>* uses. It is sent to the Diameter peers in the CER and CEA messages. The peer uses it to determine which requests are routed to this Radiator instance. *OriginRealm* must be specified.

### 3.3.9. DestinationHost

This string defines the value for *Destination-Host* for Diameter requests. The usage of this parameter depends on the Diameter application that uses this *<DiaPeerDef>*. This is an optional parameter.

### 3.3.10. DestinationRealm

This string defines the value for *Destination-Realm* for Diameter requests. The usage of this parameter depends on the Diameter application that uses this *<DiaPeerDef>*. This is an optional parameter.

### 3.3.11. SupportedVendorIds

This is an optional parameter, which defines the supported vendor IDs announced in CER and CEA messages. This has no default value and the supported vendor ID is not announced by default. The default dictionary or the configured dictionary file consist an alias group *DictVendors* for all supported vendors.

#### Example

```
# Advertise Open System Consultants and 3GPP
SupportedVendorIds 9048, 3GPP
```

### 3.3.12. AuthApplicationIds

This is an optional parameter, which defines the *Auth-Application-Id* attributes announced in the CER and CEA messages. The *Auth-Application-Id* is not announced by default.

#### Example

```
# Advertise Diameter Credit Control and EAP applications
AuthApplicationIds 4, Diameter-EAP
```

### 3.3.13. AcctApplicationIds

This is an optional parameter, which defines the *Acct-Application-Id* attributes announced in the CER and CEA messages. The *Acct-Application-Id* is not announced by default.

#### Example

```
AcctApplicationIds Base Accounting
```

### 3.3.14. VendorAuthApplicationIds

This is an optional parameter, which defines the authentication *Vendor-Specific-Application-Id* attributes announced in the CER and CEA messages. The *Vendor-Specific-Application-Id* is not announced by default. The parameter value is a comma-separated list of **vendor:application** values. Both names and direct numeric values are accepted.

## Example

```
VendorAuthApplicationIds 3GPP:3GPP-Rx, 3GPP:3GPP-Gx
```

### 3.3.15. VendorAcctApplicationIds

This is an optional parameter, which defines the accounting *Vendor-Specific-Application-Id* attributes announced in the CER and CEA messages. The *Vendor-Specific-Application-Id* is not announced by default. The parameter value is a comma-separated list of **vendor:application** values. Both names and direct numeric values are accepted.

## Example

```
VendorAcctApplicationIds OSC:Example accounting app
```

### 3.3.16. Initiator

This is an optional flag, which defines if the Radiator instance can act as a connection initiator. It is not set by default.

*Initiator* must be set if Radiator instance has to act as an initiator and create a connection to the Diameter peer defined by this <*DiaPeerDef*>. If *Initiator* is not set, the Radiator instance does not initiate connections but other instances, such as ePDG (Evolved Packet Data Gateway), must act as a initiator.

### 3.3.17. Peer

This parameter defines the name or IP address of the Diameter peer. Both IPv4 and IPv6 addresses are supported. This parameter is required when <*DiaPeerDef*> is configured to act as an initiator.

### 3.3.18. Port

This is an optional parameter, which defines the network port <*ServerDIAMETERTelco*> listens to for connections from Diameter peers. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <ServerDIAMETER>.

### 3.3.19. SCTPPeer

This parameter specifies one host name or address of an SCTP peer to connect to. An address can be an IPv4 or IPv6 address. Multiple *SCTPPeer* parameters are supported. When *SCTPPeer* is defined, it is used instead of *Host* or *Peer* parameters. Special formatting characters are supported. If SCTP multihoming is not supported, connection is attempted to each peer at a time.

When SCTP multihoming is supported, connection is attempted to all peers at once. In this case, all addresses defined with *SCTPPeer* must be either IPv4 or IPv6 addresses

Here is an example of using *SCTPPeer*:

```
# Peer has multiple IPv6 addresses
SCTPPeer 2001:db8:1500:1::a100
SCTPPeer 2001:db8:1500:2::a100
```

### 3.3.20. LocalAddress and LocalPort

These parameters control the address and optionally the port number used for the client source port, although this is usually not necessary. *LocalPort* is a string, it can be a port number or name. It binds the local port if

*LocalAddress* is defined. If *LocalPort* is not specified or if it is set to **0**, a port number is allocated in the usual way.

When SCTP multihoming is supported, multiple comma separated addresses can be configured. All addresses defined with *LocalAddress* must be either IPv4 or IPv6 addresses.

```
LocalAddress 203.63.154.29
LocalPort 12345
```

### 3.3.21. Protocol

This is an optional parameter, which allows choosing transport layer protocol, TCP or SCTP, for carrying Diameter messages. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <ServerDIAMETER>.

### 3.3.22. DisconnectTraceLevel

This optional parameter specifies log trace level for peer initiated disconnects. The default value is error level **0**. When connections are known to be short-lived, a non-default value may be useful. This parameter is available for all Stream based modules, such as *<ServerDIAMETER>* and *<AuthBy RADSEC>*.

```
# Debug logging is enough for peer disconnects
DisconnectTraceLevel 4
```

### 3.3.23. TLS_*

These parameters enable and configure of TLS (Transport Layer Security) authentication and encryption. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section "TLS configuration". To enable TLS, you need to define *TLS_Protocols* configuration parameter with the other TLS related parameters, such as certificates, that depend on your operating environment.

**Note**

Old configuration parameters *UseTLS* and *UseSSL* are obsolete and should not be used. Use *TLS_Protocols* instead.

## 3.4. <IntegratorGBA>

This section describes the configuring parameters of *<IntegratorGBA>*. Integrator is an interface between NGINX web server and Radiator. HTTP Ub request is translated into Diameter Zh request or to Diameter Zn request correspondingly.

### 3.4.1. Identifier

This is an optional parameter, which defines the name of the specific *<IntegratorGBA>* clause and its configuration.

### Example

```
Identifier nginx-bsf
```

### 3.4.2. AddToRequest

This string defines any number of RADIUS attributes to the RADIUS requests generated by *<IntegratorGBA>*. The attributes can be used for tagging the authentication requests. This is an optional parameter.

### 3.4.3. BSFDefaultLifetime

This parameter defines the default lifetime for BSF in seconds. Default value is 86400.

### 3.4.4. BSFServerName

This parameter defines the names for BSF servers. This parameter can have several values, separated by spaces or commas. If there are more than one value defined, the Zn request is sent to all BSF servers and the first successful reply is used to authenticate the Ua request.

#### Example

```
BSFServerName bsf1.ims.mnc001.mcc001.pub.3gppnetwork.org bsf2.ims.mncXXX.mccYYY.pub.3gppnetwork.
```

### 3.4.5. HSSServerName

This parameter defines the names for HSS or DRA servers. It can have several values, separated by spaces or commas. If more than one name is defined, the Zh request is sent to the first HSS or DRA server, to which Radiator has a Diameter peering up.

#### Example

```
HSSServerName dra1.mnc001.mcc001.3gppnetwork.org dra2.mncXXX.mccYYY.3gppnetwork.org
```

### 3.4.6. OriginHost

This string defines the name that <*ServerDIAMETERTelco*> uses to identify itself to the Diameter peers. It is sent to the Diameter peers in the Diameter MAR (Multimedia-Auth-Request) and MAA (Multimedia-Auth-Answer) messages. The Diameter peers use *OriginHost* to determine whether they have connected to the correct peer. *OriginHost* must be specified.

#### Example

```
OriginHost bsf.ims.mnc001.mcc001.pub.3gppnetwork.org
```

### 3.4.7. OriginRealm

This string defines the name of the Realm the <*ServerDIAMETERTelco*> uses. It is sent to the Diameter peers in the MAR and MAA messages. The peer uses it to determine which requests are routed to this Radiator instance. *OriginRealm* must be specified.

#### Example

```
OriginRealm ims.mnc001.mcc001.pub.3gppnetwork.org
```

### 3.4.8. DestinationHost

This string defines the value for *Destination-Host* for Diameter requests. The usage of this parameter depends on the Diameter application that uses this <*DiaPeerDef*>. This is an optional parameter.

### 3.4.9. DestinationRealm

This string defines the value for *Destination-Realm* for Diameter requests. The usage of this parameter depends on the Diameter application that uses this <*DiaPeerDef*>. This is an optional parameter.

### 3.4.10. SockPath

This parameter defines the path to Unix domain socket that is used for communicating with NGINX.

#### Example

```
SockPath /var/run/radiator/sock/radiator-bsf.sock
```

### 3.4.11. RewriteUsername

This is an optional parameter. It enables you to alter the username in Diameter Zh MAR.

#### Example

```
RewriteUsername s/^(XXXYYY[0-9]+)\@operator\.com\z/$1\@ims.mncXXX.mccYYY.3gppnetwork.org/is
```

## 3.5. <ServerDIAMETERTelco>

This section describes the configuring parameters of *<ServerDIAMETERTelco>*.

### 3.5.1. Port

This is an optional parameter, which defines the network port *<ServerDIAMETERTelco>* listens to for connections from Diameter peers. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <ServerDIAMETER>.

### 3.5.2. Clients

This parameter defines the IP addresses of permitted clients. If not defined, all clients are permitted, subject to authentication. The parameter value is a list of comma- or space-separated IP addresses. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <ServerDIAMETER>.

### 3.5.3. BindAddress

This is an optional parameter, which defines one or more network interface addresses that are listened to for incoming Diameter connections. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <ServerDIAMETER>.

### 3.5.4. MaxBufferSize

This is an optional parameter, which defines the maximum number of octets buffered in output. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <ServerDIAMETER>.

### 3.5.5. Protocol

This is an optional parameter, which allows choosing transport layer protocol, TCP or SCTP, for carrying Diameter messages. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <ServerDIAMETER>.

### 3.5.6. ReadTimeOut

This is an optional parameter, which defines the maximum time, in seconds, to wait for incoming Diameter connection to complete the initial handshaking. The default value is **`10`**. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section <ServerDIAMETER>.

### 3.5.7. DisconnectTraceLevel

This optional parameter specifies log trace level for peer initiated disconnects. The default value is error level **`0`**. When connections are known to be short-lived, a non-default value may be useful. This parameter is available for all Stream based modules, such as <*ServerDIAMETER*> and <*AuthBy RADSEC*>.

```
# Debug logging is enough for peer disconnects
DisconnectTraceLevel 4
```

### 3.5.8. TLS_*

These parameters enable and configure of TLS authentication and encryption. For more information, see Radiator reference manual [https://files.radiatorsoftware.com/radiator/ref.pdf] under section "TLS configuration". To enable TLS, you need to define *TLS_Protocols* configuration parameter with the other TLS related parameters, such as certificates, that depend on your operating environment.

**Note**

Old configuration parameters *UseTLS* and *UseSSL* are obsolete and should not be used. Use *TLS_Protocols* instead.

# 4. Configuring OpenResty/NGINX

This section describes the relevant configuration parameters of OpenResty/NGINX. For more information about NGINX, see NGINX web site [https://www.nginx.com/].

See `goodies/nginx-bsf-ap-combined.conf` for a configuration sample.

## 4.1. ngx_ap_server_name, global_osc_ngx_ap_server_name

This parameter defines the AP name. This is used as the NAF name when calculating NAF keys. Therefore, it must match with the hostname the Ub clients are using.

Global variable should only be defined when TLS-PSK is used and even then it is only used if a client does not supply a TLS-SNI.

### Example

```
set $ngx_ap_server_name 'xcap.ims.mncXXX.mccYYY.pub.3gppnetwork.org';
```

## 4.2. ngx_ap_socket_name, global_osc_ngx_ap_socket_name

This parameter defines the name for the Unix domain socket, which is used for communicating with Radiator.

Global variable should only be defined when TLS-PSK is used.

### Example

```
set $ngx_ap_socket_name 'unix:/var/run/radiator/sock/radiator-naf.sock';
```

## 4.3. ngx_bsf_server_name

This parameter defines the BSF name. It is used as a realm in B-TID (Bootstrapping Transaction Identifier) user names.

### Example

```
set $ngx_bsf_server_name 'bsf.ims.mncXXX.mccYYY.pub.3gppnetwork.org';
```

## 4.4. ngx_bsf_socket_name

This parameter defines the name for the Unix domain socket, which is used for communicating with Radiator.

### Example

```
set $ngx_bsf_socket_name 'unix:/var/run/radiator/sock/radiator-bsf.sock';
```

## 4.5. ngx_btid_username_realm, global_osc_ngx_btid_username_realm

This parameter defines the BSF realm suffix. Authentications are allowed only when username ends with configured suffix, leave empty to allow all.

Global variable should only be defined when TLS-PSK is used.

### Example

```
set $ngx_btid_username_realm 'mncXXX.mccYYY.pub.3gppnetwork.org';
```

This setting allows the both types of realms to be used:

```
123456789012345@bsf.mncXXX.mccYYY.pub.3gppnetwork.org
123456789012345@bsf.ims.mncXXX.mccYYY.pub.3gppnetwork.org
```

## 4.6. ngx_default_lifetime, global_osc_ngx_default_lifetime

This parameter defines the default lifetime, in seconds, for BSF.

Global variable should only be defined when TLS-PSK is used.

### Example

```
set $ngx_default_lifetime '86400';
```

## 4.7. ngx_gsid, global_osc_ngx_gsid

This parameter defines the NAF type.

Global variable should only be defined when TLS-PSK is used.

The following options are available:

- **0**

  Unspecified service

- **2**

  AP (Authentication Proxy)

### Example

```
set $ngx_gsid '0';
```

## 4.8. ngx_impi_username_realm

This parameter defines the IMPI realm suffix.

### Example

```
set $ngx_impi_username_realm 'mncXXX.mccYYY.3gppnetwork.org';
```

This setting allows the both types of realms to be used:

```
123456789012345@mncXXX.mccYYY.3gppnetwork.org
123456789012345@ims.mncXXX.mccYYY.3gppnetwork.org
```

## 4.9. ngx_notfound_cache_time, global_osc_ngx_notfound_cache_time

This parameter defines how long (in seconds) to cache failed username lookups (username not found).

Global variable should only be defined when TLS-PSK is used.

### Example

```
set $ngx_notfound_cache_time '300';
```

## 4.10. ngx_radiator_lib_path, global_osc_ngx_radiator_lib_path

Path to a shared library implementing additional TLS functions (requires TLS-PSK support from OpenResty/NGINX).

Global variable should only be defined when TLS-PSK is used.

See also `nginx_tls_lib/README`

### Example

```
set $ngx_radiator_lib_path '/opt/radiator/radiator-gba-bsf/nginx_tls_lib/libradiator_nginx_tls.s
```

## 4.11. ngx_radiator_socket_timeout, global_osc_ngx_radiator_socket_timeout

This parameter defines the time to wait for a reply from Radiator in milliseconds.

Global variable should only be defined when TLS-PSK is used.

### Example

```
set $ngx_radiator_socket_timeout '1000';
```

## 4.12. ngx_rewrite_impu_regexp

This optional parameter defines regular expression to rewrite XCAP request URI. This parameters defaults to not set. Rewrite is only required by special configurations.

### Example

```
set $ngx_rewrite_impu_regexp '(sip|tel):[0-9]{15}@[^/]+';
```

## 4.13. ngx_server_support_gba_digest, global_osc_ngx_server_support_gba_digest

This parameter defines whether BSF server supports GBA_Digest.

Global variable should only be defined when TLS-PSK is used.

The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $ngx_server_support_gba_digest 'true';
```

## 4.14. ngx_server_support_gba_js, global_osc_ngx_server_support_gba_js

This parameter defines whether NAF server supports GBA JS.

Global variable should only be defined when TLS-PSK is used.

The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $ngx_server_support_gba_js 'true';
```

## 4.15. ngx_server_support_gba_tls, global_osc_ngx_server_support_gba_tls

This parameter defines whether NAF server supports TLS. This is required for TLS-PSK and GBA JS.

Global variable should only be defined when TLS-PSK is used.

The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $ngx_server_support_gba_tls 'true';
```

## 4.16. ngx_server_support_gba_tls_psk, global_osc_ngx_server_support_gba_tls_psk

This parameter defines whether NAF server supports TLS-PSK.

Global variable should only be defined when TLS-PSK is used.

The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $ngx_server_support_gba_tls_psk 'true';
```

## 4.17. ngx_server_support_gba_uicc, global_osc_ngx_server_support_gba_uicc

This parameter defines whether NAF server supports GBA UICC.

Global variable should only be defined when TLS-PSK is used.

The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $ngx_server_support_gba_uicc 'false';
```

## 4.18. ngx_server_support_tmpi

This parameter defines whether BSF server supports TMPI usernames. The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $ngx_server_support_tmpi 'false';
```

## 4.19. ngx_server_support_zx_auth, global_osc_ngx_server_support_zx_auth

This parameter defines whether NAF server supports Ericsson Zx authentication.

Global variable should only be defined when TLS-PSK is used.

The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $ngx_server_support_zx_auth 'false';
```

## 4.20. ngx_zx_auth_scheme

This parameter defines Zx authentication scheme.

The following options are available:

- **`Digest`**
- **`SSO`**

### Example

```
set $ngx_zx_auth_scheme 'Digest';
```

## 4.21. gba_maintenance

BSF/AP maintenance mode (HTTP 503 Service Unavailable). When this parameter is enabled, "503 Service Unavailable" is always returned which will trigger a fallback in UEs trying to use the service.

The following options are available:

- **`1`**

  Enabled

- **`0`**

  Disabled

### Example

```
set $gba_maintenance '0';
```

## 4.22. xml_modification

This parameter defines whether the XML payload modification is enabled when AP sends a 200 OK response message to the UE. The currently available modification applies to all XML rulesets in the XML payload that have rules with both audio and video media elements. For example call forwarding and call barring XML rulesets can have rules with audio and video media elements, so when this parameter is enabled each rule with audio media element is moved after the corresponding rule with video media element. This modification is done with Lua in `lua/lib/xml/rule_modify.lua`.

This parameter can only be set for AP configuration and it requires that parameter *body_filter_by_lua_file* is specified. This parameter is optional.

See also Radiator GBA/BSF whitepaper [https://files.radiatorsoftware.com/radiator/whitepapers/GBA-BSF-whitepaper.pdf] section *Authentication procedure with GBA*.

The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $xml_modification 'false';
```

```
body_filter_by_lua_file /opt/radiator/radiator-gba-bsf/lua/body_filter_handler_ap.lua;
```

## 4.23. reject_when_barring_video

This parameter defines whether the XML payload check for HTTP PUT request is enabled and executed before the request is forwarded to AS. If the XML payload has barring rule with video media element, then the request is not forwarded to AS and instead a HTTP 409 (conflict) is returned to UE. This check is done with Lua in `lua/rewrite_handler_ap.lua` and `lua/lib/xml/barring_reject.lua`.

This parameter can only be set for AP configuration and it requires that parameters *rewrite_by_lua_file* and *body_filter_by_lua_file* are specified.

This parameter is optional.

See also Radiator GBA/BSF whitepaper [https://files.radiatorsoftware.com/radiator/whitepapers/GBA-BSF-whitepaper.pdf] section *Authentication procedure with GBA*.

The following options are available:

- **`true`**

  Enabled

- **`false`**

  Disabled

### Example

```
set $reject_when_barring_video 'false';
```

```
rewrite_by_lua_file /opt/radiator/radiator-gba-bsf/lua/rewrite_handler_ap.lua;
```

```
body_filter_by_lua_file /opt/radiator/radiator-gba-bsf/lua/body_filter_handler_ap.lua;
```

# 5. Abbreviations

**Authentication and Key Agreement**

AKA (Authentication and Key Agreement)

Acronym: **AKA**

**Application Proxy**

AP (Application Proxy)

Acronym: **AP**

**Authentication Proxy**

AP (Authentication Proxy)

Acronym: **AP**

**Bootstrapping Info Answer**

BIA (Bootstrapping Info Answer)

Acronym: **BIA**

**Bootstrapping Info Request**

BIR (Bootstrapping Info Request)

Acronym: **BIR**

**Bootstrapping Server Functionality**

BSF (Bootstrapping Server Functionality)

Acronym: **BSF**

**Bootstrapping Transaction Identifier**

B-TID (Bootstrapping Transaction Identifier)

Acronym: **B-TID**

**Capabilities Exchange Answer**

CEA (Capabilities Exchange Answer)

Acronym: **CEA**

**Capabilities Exchange Request**

CER (Capabilities Exchange Request)

Acronym: **CER**

**Diameter Routing Agent**

DRA (Diameter Routing Agent)

Acronym: **DRA**

**Evolved Packet Data Gateway**

ePDG (Evolved Packet Data Gateway)

Acronym: **ePDG**

**General Bootstrapping Architecture**

GBA (General Bootstrapping Architecture)

Acronym: **GBA**

**General Bootstrapping Architecture/Bootstrapping Server Functionality**

GBA/BSF (General Bootstrapping Architecture/Bootstrapping Server Functionality)

Acronym: **GBA/BSF**

**Home Subscriber Server**

HSS (Home Subscriber Server)

Acronym: **HSS**

**IP Multimedia Private Identity**

IMPI (IP Multimedia Private Identity)

Acronym: **IMPI**

**Multimedia-Auth-Answer**

MAA (Multimedia-Auth-Answer)

Acronym: **MAA**

**Multimedia-Auth-Request**

MAR (Multimedia-Auth-Request)

Acronym: **MAR**

**Network Application Function**

NAF (Network Application Function)

Acronym: **NAF**

**Transport Layer Security**

TLS (Transport Layer Security)

Acronym: **TLS**

**User Security Settings**

USS (User Security Settings)

Acronym: **USS**