EFM DGLux5 Configuration Guide

Kinetic - Edge & Fog Processing Module (EFM) 1.6.0

Revised: February 25, 2019

Table of Contents

Introduction

The Cisco Edge and Fog Processing Module (EFM) allows you to create a reliable data communications messaging system on top of your data networking infrastructure. This system provides data delivery and allows you to rapidly deploy applications, where needed, that can be at the edge or fog or in the data center. The EFM is an open platform that allows for the addition of micro services or applications by anyone, allowing for unlimited capability and growth by adding software components that optimize the results of the application, system, or outcome.

The EFM addresses the complexity of building an enterprise-ready scalable data messaging system upon which one or many applications can reside. The EFM comes with a series of tools to manage the system, the EFM system administrator, and the EFM system monitor.

Features and functions

The system's key capabilities include:

- A framework for edge and fog processing. High performance.
- Reusable micro services for collecting data from, and providing control over, devices and machines, as well as processing the data prior to delivery to its destination.
- Different options for reliable transport of data through the system, encompassing both batch and real-time streaming options.
- Flexible mechanisms for integration with IT systems, reporting, and analytics.
- An architectural framework to extend fog processing to multiple tiers: east west (fog to fog) and north south (hierarchical processing leveraging network topology).
- Easy-to-use GUI tools to simplify development, deployment, and operation for all aspects of the system.
- A pervasive control paradigm and flow of information back to micro services, devices and machines for management, control, optimization, and specific actions.
- A completely open and polyglot system where third parties can provide devices, processing storage, software modules, analytics, applications, or any combination thereof.

This is the technology that makes IoT possible, and leads to faster industry adoption of the IoT vision.

The Edge and Fog Processing Module components

Component	Description
EFM Message Broker	A small footprint component working with other brokers to form a message bus. The EFM Message Broker provides reliable and flexible data delivery between devices and micro services. The sources can be devices such as sensors or other micro services. Consumers can be micro services or user applications.
EFM C++ Broker	A multi-threaded high-performance message broker with very low footprint in order to leverage the multi-core capability of different platforms.
EFM Data Flow Editor	Defines message paths between devices and micro services.
EFM Data Flow Engine	Executes message paths between devices and micro services. We recommend installing this adjacent to the EFM Message Broker to perform data transformation and input sources that are not in the canonical data format of the system.
EFM System Administrator	Configures and manages the message broker and micro services.
EFM System Monitor	A standalone tool for operators to obtain real-time functional status of a deployed solution.
Cisco ParStream (Historian database)	Purpose-built database to handle the massive volumes and high velocity of IoT data, as well as analytics at the Edge.
EFM Tools Runtime Engine	A standalone runtime tool for visualizing dashboards and driving EFM System Administrator, EFM Data Flow Engine, and EFM System Monitor.
Links	DQL System Dataflow ParStream JDBC Device Simulator Alarming
Smart License Agent Tool for Nodes	A client that allows system users to manage license registration for Node Product IDs.
Smart License Agent Tool for Devices	The Smart License Agent client that allows system users to manage license registration for Device Product IDs.
Asset Manager	Detects and manages devices throughout the EFM Messaging system.
User Management	The User management is a component that allows managing users, groups, permissions and roles for the EFM Web based components.
DGLux5	Data Visualization Tool

Hardware requirements

EFM Message Broker EFM Data Flow Engine DQL Link System Link ParStream Link	Red Hat Linux 7, CentOS 7, 1GB RAM, Windows 2016 Server, 10 GB HD – Recommended on the same system/VM
EFM C++ Broker	Red Hat Linux 7, CentOS 7 or Ubuntu 16.04
EFM Data Flow Editor	Automatically installs with EFM Message Broker and EFM Tools Runtime Engine. Access via a web browser
EFM System Administrator	Project installs on the same system as the EFM Message Broker and EFM Tools Runtime Engine. Accessed via a web browser
EFM System Monitor	Project installs on the same system as the EFM Message Broker and EFM Tools Runtime Engine. Accessed via a web browser
Cisco ParStream (Historian Database)	Red Hat Linux 7, CentOS 7 , 6 CPU cores with 2GB RAM per core, 500 GB HD
EFM Tools Runtime Engine	Installs with EFM Message Broker
Smart License Agent Tool	Redhat Linux 7, CentOS 7, with 1GB RAM, 10 GB HD.
EFM Asset Manager	4GB RAM, 10 GB HD – Recommended on the same system/VM
User Management	Installs on the same system as the EFM Message Broker and EFM Tools Runtime Engine. Accessed via a web browser.
DGLux5	Red Hat Linux 7, CentOS 7, 1GB RAM, Windows 2016 Server, 10 GB HD

DGLux5 components' protocols and ports

The protocols and ports used by the DGLux5. The port values are configurable after installation.

TCP Port No.	Description
9080 and 9443	DGlux5 message broker and user interface port defined in the server.json configuration file. See the EFM Message Broker server.json configuration file for parameter details.

Additional ports post installation

The EFM DSLinks are microservices. They may expose additional protocol ports that are listening for incoming connections. It is necessary to verify these ports in the DSLink documentation and configure the host to allow the incoming connections as desired. For example, the MQTT DSLink provides an optional server that can listen on port TCP 8443 to MQTT clients. If the MQTT Server DSLink functionality is desired, the host must allow for the proper firewall access for this to receive the connections.

Secure Mode operation

The EFM can operate in Secure Mode to enhance the security features available for the EFM message broker and web server. Secure Mode provides these following enhacements:

- HTTPS Strict Transport Security (HSTS), which automatically redirects inbound http connections to https for message broker and web traffic
- System dslink cannot execute “system command”
- Login page won't allow browser to remember password
- Prevents the pages from being embedded in iframes
- Prevents the command action that allows shell exection by the System Link

Secure Mode is configured by indicating Y(es) during the installation of the message broker or placing the hidden file “.secureMode” in the dglux_server directory.

Using Secure Mode HSTS only affects inbound connections; outbound http and https connections are still supported.

Securing the Installation

Every install of a Cisco Kinetic EFM instance will have to meet specific requirements for performance and security. It is generally advisable, to configure the underlying platform Linux OS as tight as possible by minimizing the number of amount and privileges of processes running and services offered. Suggested is adherence to general hardening guidelines as provided by the NSA hardening guide collection at https://www.nsa.gov/ or platform specific formulations. To enable educated decisions, when the grade of security impacts performance, and where to strike a balance acceptable for the local install, the sections in this guide offer helpful information and relations.

For additional information on hardening the underlying operating system some additional references are:

Red Hat: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/pdf/Security_Guide/Red_Hat_Enterprise_Linux-6-Security_Guide-en-US.pdf

NSA hardening guide - https://www.nsa.gov/what-we-do/research/selinux/ for information on Security Enhanced Linux. See also SELinux SELinux in the RedHat Enterprise Security Guide above.

General concepts

Defining an EFM Administrator for DGLux5

It is important to note that we do not define default username and passwords for DGLux5. The first user that is defined at install becomes the administrator. After the install, using the the “users.sh” (in the $EFM_ROOT`/dglux_server/bin/ folder)` script in to additional users may be added. At least one user requires administrative priviledges for that node.

Defining a non-root Linux account for installation and operation

As a Linux security best practice, the installation RPM creates a non-root account for installing and operating the EFM. If the account does not exist, it will create the user “dglux” and group “dglux”. All examples throughout the documentation will reference this user name.

Download and Unzip the EFM package.

The software should be downloaded from CCO at www.cisco.com under “Support and Downloads.”

Unzip the image:

```
$ unzip EFM-1-6-0.zip
```

Change into unzipped folder:

```
$ cd EFM-1-6-0
```

EFM Component Installation Instructions for CentOS/RHEL

The EFM components for CentOS7 and RHEL7 are delivered in the form of RPM packages. The RPM packages are installed, updated and uninstalled using the YUM tool. During the installation of the RPM packages using YUM, third party dependencies are checked and if required downloaded and installed.

EFM RPM packages are delivered as a ZIP archive containing all RPM packages. This archive has to be installed on the target system before the RPM packages it contains can be installed with YUM.

Install local EFM repository of RPMs

Copy EFM archive from corresponding platform subfolder of the EFM-1-6-0.zip file to the target system.

The archive name follows the format repo-<major-version>.<minor-version>.<patch-version>-<build-number>.zip, i.e. " repo-1.6.0-69.zip"

The following steps must be carried out for this as user 'root':

Step 1	`$ unzip repo-1.6.0-69.zip`	Unpack the archive in the efm user home directory. This creates the directory " /repo" , which contains all EFM RPM packages

Step 2	`cat << EOF > /etc/yum.repos.d/efm.repo`
	`[local]`
	`name=EFM Repository`
	`baseurl=file:///home/efm/EFM-1-6-0/centos7/repo`
	`gpgcheck=0`
	`enabled=1`
	`EOF`
		Register the local EFM archive for yum. Therefore create a text file with the following content under the path " /etc/yum.repos.d/efm.repo" (the directory " /etc/yum.repos.d" already exists)
		If you used a different folder in Step #1, adapt the path for *baseurl* correspondingly.

Step 3	`$ yum updateinfo`	Update YUM in order get the latest package versions. Now yum is prepared for the installation of the EFM RPM packages.

Install EFM Component RPM packages

In order to install an EFM Component RPM package in CentOS and RHEL invoke:

```
sudo yum install -y <package-name>
```

For example:

```
sudo yum install -y efm-dglux
```

This commnd installs the efm-server in the directory `/opt/cisco/kinetic/efm_server`. A user `efm` will automatically be created and will be owner of the corresponding installation files.

`/opt/cisco/kinetic` is the common home directory for all installed EFM packages.

Each package is installed in a separate subfolder, with the exception of additional modules for the EFM server, e.g the efm-server-admin or efm-server-monitor, which will be installed below the " efm_server" folder.

Required dependencies are automatically solved, e.g. if System Administrator is installed using the command

```
sudo yum install -y efm-dglux
```

Dglux5 will also automatically be installed, if it is not already present.

Some packages also configure systemd services which can be used to start and stop the components. These services are by default disabled, but can be easily enabled and started.

Prerequisite to allow installation of Links from Git repositories

The EFM System Administrator allows for installing links from Git, or the system link in the (/sys/links/Install Link/from Git). The git command line application must be installed for this function to properly operate.

```
$ sudo yum install git
```

Installation of DGLux5

To install DGLux5 run the following command:

```
sudo yum install -y efm-dglux
```

To run EFM Server and DGLux5 Server in parallel, both need to be owned and executed by different system users. The installation process takes care of corresponding user creation and permissions. To run EFM applications and EFM Server the system user `efm` is used. To run DGLux Server the system user `dglux` is used. The user dglux is automatically created, owns the installation and is configured to run the corresponding service.

The following systemd service is automatically added:

```
efm-dglux.service
```

For improved security, no default UI user is created. To create a DGLux user go to folder

```
/opt/cisco/kinetic/dglux_server
```

and run the following command:

```
./bin/users.sh add -u [USERNAME] -p [PASSWORD]
```

If this user shall be configured as superuser the command needs to be:

```
./bin/users.sh add -u [USERNAME] -p [PASSWORD] -s
```

Starting DGLux:

```
service efm-dglux start
```

To connect DGLux5, use a web browser and proceed to

```
https://[SERVER IP ADDRESS]:9443
```

and log in with the admin user credentials you just registered.

EFM Server and DGLux5 licensing

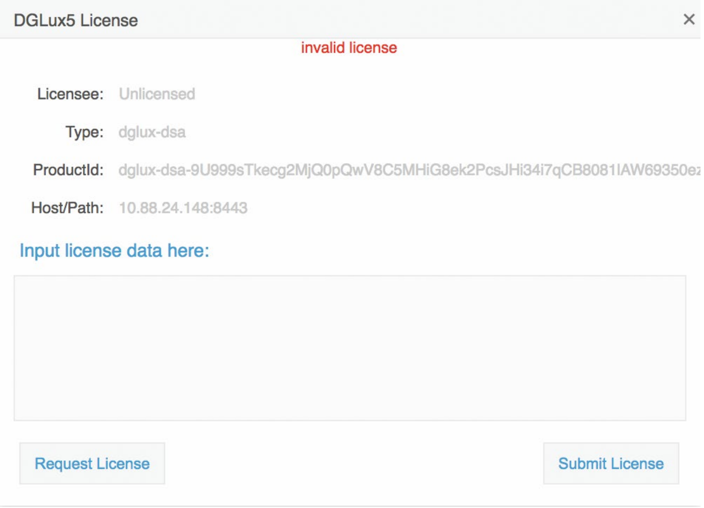
The EFM Server, upon startup, installs a .dglogik licensing file in the home directory of the efm user. This license enables the EFM to function as a unlimited node and supports the specific EFM projects: EFM System Administrator and EFM System Monitor. DGLux5 at startup also installs a .dglogik licensing file in the home directory of the dglux user.

Starting the appropriate services assures that each application is run as the correct user and licensing overlap is avoided.

The DGLux5 also requires a license to operate. The trial license expires automatically after the 6-month trial. This license is not renewable and require the purchase of a additional license for operation.

Requesting a DGLux5 6-month trial license

1. Using a web brower client, connect to the dglux port using **http://[Server IP address]: 9080** or **https://[Server IP address]:9443**.
2. Log in as administrator user and password defined at installation.



DGLux5 License

invalid license

Licensee: Unlicensed

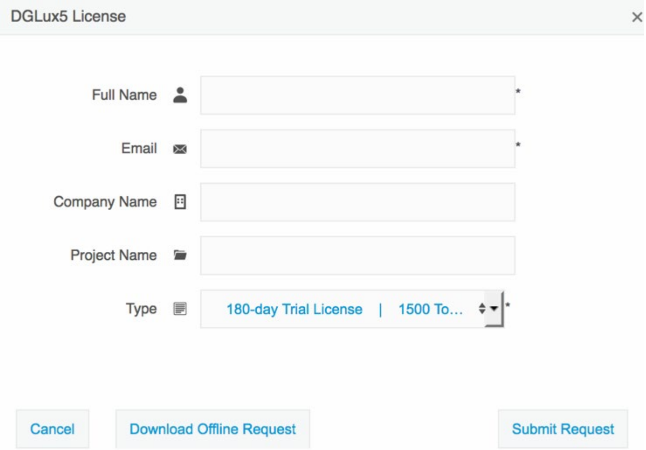
Type: dglux-dsa

ProductId: dglux-dsa-9U999sTkecg2MjQ0pQwV8C5MHiG8ek2PcsJHi34i7qCB8081IAW69350ez

Host/Path: 10.88.24.148:8443

Input license data here:

3. Click **Request License**. Complete the following form, click **180-day Trial License | 1500 Topics**, and then click **Submit Request**.



DGLux5 License

Full Name *

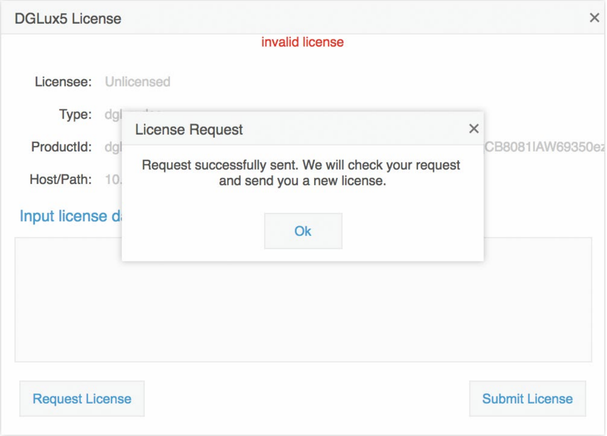
Email *

Company Name

Project Name

Type: 180-day Trial License | 1500 To... *

The following results display:



DGLux5 License

invalid license

Licensee: Unlicensed

Type: dgl

ProductId: dgl CB8081IAW69350e

Host/Path: 10.

Input license d

License Request

Request successfully sent. We will check your request and send you a new license.

In most instances, licensing will be automatic if connected to the Internet. Otherwise a license will be returned via email to the user's email address.

4. Connect to the dglux server port using **http://[Server IP address]:9080** *or* **https://[Server IP address]:9443**. Once the license has been approved, the following pop-up menu will display. If you agree to the End User License Agreement, click **I agree**.

DGLux5 Topics Definitions with DGLux5 licensing

A Topic is a subscription to a data feed with a unique path used at run-time in DGLux UI & Dataflow. There are no other restrictions, You get unlimited users, projects, widgets etc.

Unique topics only count once and can be used in multiple components without incrementing the count.

The following items can count as a topic:

- Real Time Value
- Historical Value (trended data)
- Query
- Command/Control Action to override a value

Examples :

- Single unique value bound to a gauge, a text and used in dataflow as 1 topics.
- It counts 1kB of data throughput per hour as 1 topics. Page sizes have nothing to do with it.

Additional Details:

Topics counts get reset every month. This is to ensure that the current topics count is accurate and if the user creates a page and chooses to remove it later, those topics are released.

Topics count can be manually reset by restarting the DGLux5 server.

Numeric changes to a query do not count. For example, if the user is pulling up a trend and has the functionality to select any date range the numeric changes to the query does not count as unique changes.

If topics count is exceeded, the user will not be locked out of the application, instead a watermark stating that the installation is unlicensed will appear at run-time.

The number of topics does not increase due to users, it’s only counting unique data connections.

Viewing Topics in the Dataflow Editor

Topics Actual and **Topics Preview** counts are displayed in Metrics Panel (Data->Sys->atrius)

Topics Actual – Topics that you have used against your allotted amount

Topics Preview – Lets you preview how many topics a given page is using and can be reset (right click – reset)

Topics licensed – Your allotted amount of topics

Using Newly Defined Services

The registered services can be administrated, configured and used with usual systemctl and service commands.

If a new added service is not known by the system yet, run the following command to reload the services:

```
systemctl daemon-reload
```

All services are disabled by default for security reasons. To configure the services to automatically start on reboot use the following commands:

For DGLux:

```
systemctl enable efm-dglux
```

Upgrading to EFM version 1.6.0 from an existing installation on CentOS7 and RHEL7

EFM Upgrade and Migration on CentOS7 and RHEL7 from EFM 1.5.x to EFM 1.6.0

In the following section the upgrade/migration from an existing EFM 1.5 installation to EFM 1.6.0 is described.

Prerequisite: Validate or create required users

Before you start the migration, please ensure all required system users are created with correct settings.

Username	Home Directory	Command to create user
dglux	Required	`useradd -r -U -m dglux`

Upgrade DGLux5

To upgrade DGLux5 Server run the following commands:

```
cp -r $EFM_ROOT/dglux_server /opt/cisco/kinetic/dglux_server
sudo chown -R dglux:dglux /opt/cisco/kinetic/dglux_server
sudo yum install -y efm-dglux
service efm-dglux start
```

See Installation of DGLux5 for more details on requesting a trial license if necessary.

Ubuntu no longer supported for DGLux5

On Ubuntu16.04 EFM only supports the new C++ Broker, no other components are supported on Ubuntu. It is necessary that the user migrate to RHEL7 or CentOS7 for EFM 1.6.

Troubleshooting

Linux Firewall issues

Redhat and CentOS initially is configured by default with the firewall service turned on and blocks all incoming connections. It is necessary to consult the Operating System Guide to turn off or allow only the known service ports for the EFM connections. The proper configuration needs to be defined by the host administrator.[1]

For the firewall to allow for incoming connections on the 9443 and on Redhat/CentOS, the following commands can be executed:

```
$ sudo firewall-cmd --add-port=9443/tcp --permanent
```

If using unencrypted connections to the broker on the default 8080 port, this will need to be added:

```
$ sudo firewall-cmd --add-port=9080/tcp --permanent
```

You must restart the firewall to implement the changes.

```
$ sudo firewall-cmd --reload
```

Note that if any incoming connections for DSLinks to the ParStream database, etc. exist, those specific ports should be configured to allow incoming connections.

Proxy Server challenges and the EFM Message Broker

In some environments, it might be necessary to define a proxy server to access the Internet due to security restrictions. The EFM message broker uses a localhost communication to connect to the DSLinks on the same host and usually any proxy server configuration inhibits some of this functionality from functioning properly.

We have observed in DGLux5 that some DSLinks connect to the message broker, while others do not if there is a proxy server configured.

In order to successfully connect to all the DSLinks it stopping the Message Broker be necessary, remove the proxy settings and start again the message broker. For example:

- Stop the message broker with `/opt/cisco/kinetic/dglux_server/bin/daemon.sh stop`
- Remove the proxy server settings in the environment or system configuration
- Start the message broker with `/opt/cisco/kinetic/dglux_server/bin/daemon.sh start`

[1] Unless properly configured and made permanent, on RedHat and CentOS, the firewall service will restart in the default configuration.

An alternative to removing the use of the proxy server is to define an exclusion list that includes the localhost. In this manner at least the localhost will not be forwarded to the proxy server and communications between the Message Broker and the DSLinks that are on the local host form a connection.[2]

[2] See http://xmodulo.com/how-to-configure-http-proxy-exceptions.html for examples on Linux.

Configuring the EFM Dart Message Broker or DGLux5 server configuration via the server.json file

The EFM Dart broker is configured in the server.json file. The system administrator can edit the text file. Modifications to this file should be performed when the broker is not running to avoid the content being overwritten by the message broker. The new configuration will take effect after startup.

The Dart Broker configuration file server.json are the same for EFM Dart Broker and the DGLux5.

An example server.json configuration file located in the $EFM_ROOT/dglux_server folder and does not necessarily contain all parameters:

```
{
  "debug": false,
  "host": "0.0.0.0",
  "port": 8080,
  "httpsPort": 8443,
  "certName": "cert.pem",
  "certKeyName": "key.pem",
  "certPassword": "",
  "enableHSTS": false,
  "enableCSRFProtection": false,
  "strictFileUpload": {
    "enabled": false,
    "useClamAV": false,
    "extensions": [
      "dg5",
      "dgi",
      "crt",
      "key",
      "woff",
      "ttf",
      "gif",
      "svg",
      "png",
      "jpg",
      "xml",
      "json",
      "sql",
      "csv"
    ]
  },
  "disableFileSecurity": false,
  "isAlwaysOffline": false,
  "broadcast": false,
  "workers": 1,
  "updateInterval": 200,
  "static": {
    "/.well-known": "/opt/cisco/kinetic/efm_server/.well-known"
  },
  "linkConfig": {},
  "disabledLinks": [],
  "enableUptimeChecker": true,
  "uptimeCheckUrl": null,
  "upstream": {},
  "strictTls": false,
  "quarantine": false,
  "allowAllLinks": true,
  "defaultPermission": [
    [
      ":config",
      "config"
    ],
    [
      ":write",
      "write"
    ],
    [
      ":read",
      "read"
    ],
    [
      ":user",
      "read"
    ],
    [
      ":trustedLink",
      "config"
    ],
    [
      "default",
      "none"
    ]
  ],
  "useRuntimeManager": false,
  "useDartRuntimeManager": false,
  "useJavaRuntimeManager": false,
  "passwordHasherIterations": 10000,
  "passwordHasherKeyLength": 32,
  "loginRedirectPath": "/",
  "guestLoginRedirectPath": "/assets/",
  "authType": "proxy",
  "twoFactorAuth": "none",
  "runPortChecks": true,
  "storageDriver": "simple",
  "downstreamName": "downstream",
  "loggers": [],
  "proxies": {},
  "hooks": {},
  "distributionUrl": "NO",
  "linkRepositoryUrl": "https://dsa.s3.amazonaws.com/links/links.json",
  "serverVmFlags": [],
  "userTimeout": 525600,
  "allowBrowserCaching": false,
  "serverLogLevel": "INFO",
  "enableLogCompression": true,
  "logRotationInterval": 0,
  "enableIPv6": true,
  "dartRuntimeManagerVmFlags": [],
  "javaRuntimeManagerVmFlags": [],
  "allowPasswordChanges": true,
  "keepCustomAssets": true,
  "linkManagerEnvironment": {},
  "timeHttpRequests": false,
  "generatedCertificateSubject": "/C=US/ST=California/L=Oakland/O=DGLogik
Inc./OU=Customers/CN=*",
  "enableCertificateGeneration": true,
  "alternativeBrokerUrl": null,
  "httpPathClassification": {},
  "corsProxyRules": "",
  "enableGit": false,
  "enableSingleSignOnServer": false,
  "maxQueueSize": 256,
  "ssoProviderUrl": null,
  "formatDg5": false,
  "allowedCorsRegexString": null,
  "loginAuditFileName": "audit.log",
  "loginAudit": false,
  "blockOutsideGuests": false,
  "brokerName": "broker-",
  "runBrokerInMain": true
}
```

In the following table, the default values are listed that are assumed by the server, if the parameter is not present in the server.json.

Option	Description	Default Value	Comments
allowAllLinks	When the value is true, all incoming DSLink connections will be accepted to /downstream. When the value is false, an incoming DSLink without proper authentication will be rejected unless quarantine is enabled.	true	
allowBrowserCaching	When enabled, this value will add Cache-Control headers for 300 seconds on static files such as .dg5, images, etc.	false	
allowedCorsRegexString	If you wish to allow, but restrict, the access of external sites to interface with your EFM server, you can set a Regular Expression string here which much match for the external server requests to be completed.	null	
allowPasswordChanges	When true, this value will enable passwords to be updated via the `/change_password` URL (after the user has logged in). This is only work if supported by the current (authType)[#authtype]	true	
alternativeBrokerUrl	If you wish for all DSLink connections to be forwarded to a separate broker rather than the default broker, you would specify the URI of the alternative broker here. This was primarily used for legacy installations.	null	
authType	Determines the authentication provider to use.	file	
blockOutsideGuests	Enable this value if you wish to require a valid user login to view all projects.	false	

broadcast	When this value is true, the server's broker is broadcast to the local network for discovery by other machines. When this value is false, the broadcast service is not enabled.	true	
certKeyName	SSL private key file name. Leave blank to disable HTTPS		
certName	SSL certificate file name. Leave blank to disable HTTPS		
certPassword	SSL certificate password. Set to null to disable HTTPS		
corsProxyRules	The EFM server may also be used to proxy requests to external servers. To limit the locations which the proxy can access, a list of addresses, separated by new lines, may be added to the string.	" "	
dartRuntimeManagerVmFlags	When the useRuntimeManager or useDartRuntimeManager options are enabled and the platform supports the use of a runtime manager, then the flags provided here are passed to the Dart VM prior to starting the DSLink manager.	[]	
debug	Enable/Disable Debugging Mode	false	For production site, this should always be false, debug:true may result in memory leak and bugs.

defaultPermission	Default permission setting for the root node. When this value is null, permissions are disabled, and everything has the config permission.	[[" :config" , " config"], [" :write" , " write"], [" :read" , " read"], [" :user" , " read"], [" :trustedLink" , " config"], [" default" , " none"]]	
disableFileSecurity	When this value is true, then any user can access any file. When this is false, file permissions are checked.	false	
distributionURL	This value is the url used to check for updates of the EFM server. This value can be managed in the `/sys/config` nodes (generally should not change from default).	NO	
downstreamName	This value is the name of the downstream connections node. Previously releases used a downstream name of `conns`. However it is recommended to leave this as the default value, as other Requester DSLinks may make assumptions of the correct path.	downstream	
enableCertificateGeneration	When this option is set to true, the server will attempt to generate self-signed SSL certificates prior to launching the server. This will set the appropriate certName, certKeyName. If these values are not empty, then certificate generation will be skipped.	true	
enableCSRFProtection	When this value is true, the HTTP server will add specific headers and cookies to help mitigate Cross-Site Request Forgery attacks.	false	

enableGit	This value will enable git version control over your project directory. When enabled, modifications to files in the project will be committed to a git repository at the same file path, and can be used in project management to see a history of changes and even revert changes.	false	
enableHSTS	When this value is true, the HTTP server will always redirect to the HTTPS server, and the HTTPS server will have HSTS enabled to route requests automatically to the HTTPS server.		
enableIPv6	Toggles support for IPv6 connections	false	
enableLogCompression	If this value is true, then when log files reach approximately 5MB in size, they will be rotated and compressed. Log files will be renamed to `<logfile>.<millisecondTimeStamp>.gz`	true	
enableSingleSignOnServer	In an environment where there are multiple instances of the EFM Server installed on the network, it is possible to allow all instances to refer to one primary server for authentication. When this option is enabled, this server will act as a primary server and allow other EFM Server instances to query this server for a user session on this server and if found, share it with the other instance. This requires the ssoProviderUrl be supplied to the other EFM Server instances.	false	
enableUptimeChecker	The server also comes with a checker which will periodically check to verify that the server is still up and running and responsive. Setting this value to false will disable the uptime checker.	true	
formatDg5	When this value is true, EFM client will save dg5 in a formatted and json with key sorted, makes it easy to track changes.	false	
generatedCertificateSubject	If the option enableCertificateGeneration is enabled, this is the subject used when generating the self-signed certificate.	/C=US/ST=California/L=Oakland/O=Acuity Brands Inc./OU=Customers/CN=*	
guestLoginRedirectPath	Determines the URI that a user is redirected to when login is complete.	/	
hooks	This value is designed to execute a specific command line program at various server states. Currently the only supported state is `ready` which executes when the server has finished loading. The Map contains keys of state (eg ready) and value of a list of command line programs to execute.	{}	

httpPathClassification	This value is a map of paths which may match a specific classification string. The key is the classification and the value is a list of paths which match that classification. If enabled and a requested path to the server matches a path in that classification, then that request will be treated as that type of classification request even if not matching the original hardcoded path. Currently the only supported classification is `session`.	{}	
httpsPort	HTTPS port to listen on. If this is less than or equal to 0, and/or certName or certPassword is not provided, then the server does not listen on any port for HTTPS. Ensure that if you install a custom certificate, you fill in the certName, certKeyName and certPassword fields.	8443	At least one of port or httpsPort must have a valid port number assigned.
isAlwaysOffline	Indicates that a server is expected to never have a full internet connection. This will prevent the server from trying to download the list of DSLinks available in the remote repository.	false	
javaRuntimeManagerVmFlags	When the useRuntimeManager or useJavaRuntimeManager options are enabled and the platform supports the use of a runtime manager, then the flags provided here are passed to the Java VM prior to starting the DSLink manager.	[]	
keepCustomAssets	When the value is true, custom assets in www/assets are kept upon updating EFM Server.	false	
linkConfig	Each DSLink may optionally specify its own configuration parameters to use. These configuration parameters can be see under the `/sys/links/<linkName>/configs` node. If you modify one of those parameters, the value is updated in the server configuration, as opposed to directly modifying the DSLink's configuration file. This value will vary depending on the DSLinks installed, their given names and the configuration parameters they may provide. It should be modified from the DSA node tree rather than by hand.		
linkManagerEnvironment	This value is a map of environment variables to set when the DSLink manager is started.	{}	
linkRepository	This value is the url used to check for updates for any of the DSA Links installed via repository. This value can be managed in the `/sys/config` nodes (generally should not change from default).		

loggers	The server contains a number of specialized loggers, particularly for debugging, which may be added here to retrieve verbose logging information. Some examples include " File Service" and " Execute" . These would normally be advised to be enabled at the request of support.		
loginAuditFileName	This value only applies when *loginAudit* is enabled. This will be the filename, within the `/logs` directory, in which the login audits are recorded.	audit.log	
loginAudit	When enabled, this option will log to the *loginAuditFileName* any user logins, it will record the DateTime, username, and the IP address from which the request originated. It will also log any time a user's IP address changes during an active session.	false	
loginRedirectPath	Determines the URI that a user is redirected to when login is complete.	/	
logRotationInterval	This value is the number of seconds to wait before rotating log files. When this option is enabled (anything greater than 0) enableLogCompression will not be used. After the specified period, a log file will be renamed to `<filename>.<number>` the higher the number the older the log file. Any files greater than 2 will be removed.	0	
maxQueueSize	This value is the maximum number of items stored in the queue to be sent, if the queue reaches a volume greater the behaviour will vary depending on the QOS settings (merged, dropped etc).	256	
passwordHasherKeyLength	When using file based authType, this value determines the number of bytes that the encoded password should store.	32 bytes	
passwordHasherIteration	When using file based authType, passwords are encrypted locally using PBKDF2. This value determines the number of iterations of the PBKDF2 algorithm used to encode the password.	10000	
port	HTTP Port to listen on. If this is less than or equal to 0, then the server does not listen on any port for HTTP.	8080	At least one of port or httpsPort must have a valid port number assigned.
proxies	This value is a Map of path (key) and URI (value) pairs. Requests to the path will be forwarded to the URI	{}	

quarantine	** This setting has no effect when allowAllLinks is true ** When the value is true, a new incoming DSLink without a token will be put in /sys/quarantine. A quarantined DSLink can only work as a responder. Use the /sys/quarantine/authorize to move a quarantined DSlink to /downstream.	false	
runPortChecks	When set to true, this option will verify that the configured ports for the server (HTTP and HTTPS) are valid and available for use prior to actually starting the server.	true	
serverLogLevel	Sets the log level verbosity. Levels are: `NONE`; `SEVERE`; `WARNING`; `INFO`; `FINE`; `FINEST`; `ALL`; `DEBUG`. Each level will report its level and all prior to it. (Example: INFO will log all INFO, WARNING and SEVERE messages).	INFO	
serverVmFlags	This value is a list of flags to add to the server when being started. They only apply to the EFM server and not any managed links.		
ssoProviderUrl	When this value is supplied, it must be the URI of another EFM server instance. This server will request an existing session from the supplied server and if found grant access via that session. If no session is found, the user will be prompted to log into that server and will be redirected to this instance once successfully authenticated.	null	

static	Configures a static directory mapping. This is used to serve files and directories on the server. Example:
	{
	" /static" : " /srv/http/static"
	}
		{" /.well-known" : " /path/to/dsa/dglux-server/.well-known"}	

storageDriver	This option is available for future expansion for how data is persisted at various QOS levels. Currently only simple is supported.	simple	

strictFileUpload	**strictFileUpload** is a configuration map that contains 3 fields. When this option is enabled, it will affect various file upload capabilities. Notably, it prevents guest users from being able to upload a file; It will limit uploads to only explicitly permitted file extensions; and possibly scan uploaded files for viruses.
	The configuration options are:
	enabled
	When set to true, strictFileUpload is enabled. If false, it will disable to strictFileUpload checks.
	useClamAV
	When set to true, the server will attempt to find Clam Antivirus on the system and if located, it will try to utilize this to scan any file uploads the server receives from a user. If this value is false, or the ClamAV was not found on the system, antivirus scans will be skipped, but other strictFileUpload conditions still apply if enabled.
	extensions
	An allow list of permissible file extensions (omitting the leading .) When strictFileUpload is enabled, the filename must end in one of these extensions or the upload will be rejected.
		{
		" enabled" : false,
		" useClamAV" : false,
		" extensions" : [
		" dg5" ,
		" dgi" ,
		" crt" ,
		" key" ,
		" woff" ,
		" ttf" ,
		" gif" ,
		" svg" ,
		" png" ,
		" jpg" ,
		" xml" ,
		" json" ,
		" sql" ,
		" csv"
		]
		}	

timeHttpRequests	If enabled, this value will cause all HTTP requests to the EFM Server to be timed and the log will be updated with the request and elapsed duration.	false	
twoFactorAuth	Determines the two factor authentication provider to use. Supported Two-Factor Authentication Providers • none: Don't enable two factor authentication. duo: Duo Two-Factor Authentication	none	
updateInterval	Only affects the responder. When this setting specified, a responder must not send stream updates to server more often than the minimum interval in milliseconds, value subscriptions in the responder should be cached. If a value subscription update is already cached, it must update the cache with the new value to prevent useless updates or updating an incorrect value. This value only affects the time between two updates of the same stream. If the responder does not respect the interval, the requestor might close the connection due to flooding.	200	

upstream	A list of upstream brokers that are defined with a locally referenced upstream name, locally referenced broker name, the connection URL that always has the suffix /conn, token and group.
	For example, the name "upstreamName", local name " ThisName", the URL https://192.168.22.93:443/conn,enable, no token and permission group ":config".
	`"upstreamName": {`
	`"name": "ThisName",`
	`"url":`
	`"https://192.168.22.93:443/conn",`
	`"enabled": true,`
	`"token": null,`
	`"group": ":config"`
	`}`		

uptimeCheckURL	The server has a built-in checker to verify it is still running, and restart it if it goes offline or drops connections. By default the checker will attempt to connect to localhost. However if the server is bound to a different interface in the host parameter, you will need to specify the correct URL for the server. It should end in `/ping`. This only applies when enableUptimeChecker is enabled. Example: `"https://169.254.100.100/ping"`		
useDartRuntimeManager	When the value is true, the Dart Runtime Manager is used for Dart DSLinks. The Dart runtime manager reduces resource consumption by merging Dart DSLinks into a single process.	false	
useJavaRuntimeManager	When the value is true, the Java Runtime Manager is used for Java DSLinks. The Java runtime manager reduces resource consumption by merging Java DSLinks into a single process.	false	
userTimeout	Number of minutes of user inactivity (nothing being loaded from the server) after which session times out. This is a general setting, cannot be set per user.	525600	
useRuntimeManager	This value enables both the useDartRuntimeManager and useJavaRuntimeManager on the server if applicable. **WARNING: Setting this value has no effect on a Windows based server**		

workers	Number of Server Workers. For low end devices, this should stay at 1. For large machines, this can be set up to a maximum of 128. It is recommended that you do not exceed the number of logical processors on your machine.	For single-core machines, this is 1, for other devices, this is 2.	

Obtaining documentation and submitting a service request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS Version 2.0.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.