Secure System Configuration File

- Feature Summary and Revision History, on page 1
- Feature Description, on page 2
- How System Configuration Files are Secured, on page 2
- Configuring Signature Verification, on page 3

Feature Summary and Revision History

Summary Data

Applicable Product(s) or Functional Area	All
Applicable Platform(s)	ASR 5500 VPC-DI VPC-SI
Feature Default	Disabled
Related Changes in This Release	Not Applicable
Related Documentation	• *ASR 5500 System Administration Guide* • *VPC-DI System Administration Guide* • *VPC-SI System Administration Guide*

Revision History

Revision Details	Release
First Introduced.	21.3

Feature Description

A system configuration file contains crucial configuration information used to setup and operate the operator's network. The configuration file must be properly authenticated before it is loaded to avoid unauthorized changes to the file that could harm the network.

This feature enables the system configuration file to be signed with an RSA key to ensure the integrity and authenticity of the configuration file before it is loaded. The operator can sign the configuration file with a private key, and the system uses a public key to validate the signed configuration file before loading it.

How System Configuration Files are Secured

Create a Digital Signature

The operator can sign the configuration file using the following steps:

1. Perform an SHA512 hash on the configuration file to create a message digest.

   Example (Linux/OpenSSL):

   ```
   openssl dgst -sha512 -binary -out digest cfg_file
   ```

2. Create a digital signature by encrypting the message digest value with the RSA private key.

   Example (Linux/OpenSSL):

   ```
   openssl pkeyutl -sign -in digest -inkey pri_key.pem -out sig \
   -pkeyopt digest:sha512 -pkeyopt rsa_padding_mode:pss \
   -pkeyopt rsa_pss_saltlen:-2
   ```

3. Convert the digital signature to a base64 format (A '#' is added at the beginning, and a new line at the end).

   Example (Linux/OpenSSL):

   ```
   echo -n "#" > sig_base64
   base64 sig -w 0 >> sig_base64
   echo "" >> sig_base64
   ```

4. Append the original configuration file with the digital signature.

   Example (Linux/OpenSSL):

   ```
   cat sig_base64 cfg_file > signed_cfg_file
   ```

Generating the Public and Private Keys

The RSA public key is stored in PEM format (.pem file), and can be generated using one of the following OpenSSL commands in the example below:

```
openssl rsa -in pri_key.pem - pubout -out pub_key.pem
--or--
openssl rsa -in pri_key.pem -RSAPublicKey_out -out pub_key.pem
```

An RSA private key in PEM format can be generated using the OpenSSL command in the following example:

```
openssl genrsa -out pri_key.pem 2048
```

For more information on the **openssl rsa** and **openssl genrsa** commands, refer their respective OpenSSL manual pages.

Validate the Digital Signature

When signature verification is enabled, validation of the digital signature occurs when the system boots up and loads the configuration file (or any time when the config file is loaded). The system determines if signature verification is enabled (or disabled) by looking for the *.enable_cfg_pubkey* file in the secure directory. For more information, refer Enable or Disable Signature Verification, on page 4.

The system validates the signed configuration file using the following steps:

1. Extract the RSA public signing key from the flash.
2. Extract the configuration file's digital signature (the first line).
3. Convert the signature from base64 to binary format.
4. Decrypt the signature using the RSA public key.
5. Calculate the SHA512 hash for the plain config file resulting in a message digest.
6. Compare the decrypted signature value and newly calculated message digest. If they match, the configuration file is successfully validated.

Configuring Signature Verification

Import RSA Public Key for Verification

To verify the signed configuration file, an RSA public key (in PEM format) must be imported. Use the following command to import the RSA public key:

Important This command can only be executed from the console.

cfg-security import public-key url *url_address*

Notes:

- Any existing .pem file will be replaced with the new .pem file when the command is executed.
- *url_address* may refer to a local or a remote file, and must be entered using the following format:

  [file:]{/flash | /usb1 | /hd-raid | /sftp}[*/directory***]/***filename*

  tftp://*host***[:***port***][/**<*directory*>**]/***filename*

  ftp://[*username***[:***password***]@]***host***[:***port***][/***directory***]/***filename*

  sftp://[*username***[:***password***]@]***host***[:***port***][/***directory***]/***filename*

  http://[*username***[:***password***]@]***host***[:***port***][/***directory***]/***filename*

  https://[*username***[:***password***]@]***host***[:***port***][/***directory***]/***filename*

Enable or Disable Signature Verification

Use the following command to enable (or disable) signature verification in the configuration file:

Important This command can only be executed from the console.

```
[ no ] cfg-security sign
```

Notes:

- Enabling signature verification (**cfg-security sign** command) will create an empty file named *.enable_cfg_pubkey* in the same directory where the PEM file exists.
- Use the **no cfg-security sign** command to disable verification of signature in the configuration file. Disabling signature verification (**no cfg-security sign** command) will remove the *.enable_cfg_pubkey* file.
- The system looks for the *.enable_cfg_pubkey* file to determine if signature verification is enabled or disabled.