

DI-Net Encryption

- Revision History, on page 1
- Feature Description, on page 1
- How it Works, on page 1
- Configuring Encryption Algorithm, on page 4
- Appendix, on page 4

Revision History

Revision Details	Release
First introduced.	21.27.4

Feature Description

The VPC-DI systems use Advanced Encryption Standard Cipher Block Chaining (AES CBC) algorithm to encrypt the traffic flowing between different cards. However, the CBC algorithm has one drawback as it uses an unauthenticated encryption mode there is a possibility of attackers tampering with the encrypted traffic at any given point in time. To avoid this issue an authenticated encryption algorithm is used which provides better protection and aids in data integrity.

The Galois or Counter Mode (GCM) encryption algorithm supports authenticated encryption mode which helps in overcoming this vulnerability. Also on the decrypting side, GCM uses Additional Authentication Data (AAD) to authenticate the payload.

How it Works

Since the GCM encryption algorithm is authenticated, it is used in the DI-Net traffic encryption process. It is highly secure, and the same Initialization Vector (IV) is never repeated for any given key value. The *param.cfg* file is used to configure the encryption algorithm.

Both Cipher Block Chaining (CBC) and GCM algorithms use block cipher and Exclusive OR (XOR) logic with distinct internal functions.

The CBC encryption process consists of XORing the previously encrypted blocks known as cipher texts with the unencrypted blocks known as plain text and then encrypting the resultant block with a block cipher. Decrypting of encrypted data or cipher text using a block cipher and by XORing the resultant block with the previous cipher text block, yields the plain text data.



Note The first block is treated as a special case as it does not belong to any previous block and uses the IV instead of the previous block data.

The GCM algorithm is a combination of counter mode encryption and authentication (CTR + Auth). It combines the Galois field multiplication with the counter mode of operation for block ciphers and aids in the conversion of block ciphers into stream ciphers. Each block is encrypted with a key stream's pseudo random value. Because of the successive increment of IV values, each block is encrypted with a unique value which is never repeated.

The Galois field multiplication component considers each block as its own finite field for encryption based on the Advanced Encryption Standard (AES) standard. The AES GCM incorporates handshake authentication with additional data authentication. Also, the GCM encryption or decryption process can also be parallelized anytime, and the built-in authentication makes it resistant to payload tampering and to paddle oracle attacks due to which it is preferred over the CBC algorithm.

AES-CBC-256

The master Control Function (CF) card generates the encrypted password using **openssl**. The CF card is solely responsible for creating the passwords and secret codes that all of the cards use during the boot-up process. All passwords have the slot numbers appended to them, during the key and IV generation process. Any card is allowed to generate the key and IV of any other card. The same process is followed for creating the dynamic IV table as well. The keys are of length 256 bits each and the IVs are of length 128 bits.

During the encryption process, the source card uses its own key whereas the IV is generated at random. The source card's corresponding IV is XORed with an IV from the dynamic IV table which is selected based on the hash function output, which includes the source and destination addresses of the IP header as well as a random number. This random number is included in the crypto header.

During the decryption process, the destination card uses the source card slot number to select the key and source address along with the destination address and a random number from the headers, before selecting the IV.

AES-GCM-256

To change the encryption algorithm to **aes-gcm-256**, the encryption function requires Additional Authentication Data (AAD) as an additional input for the encryption algorithm. It can be anything that is forwarded between the source and the destination which ensures the key and IV pair are never re-used. The GCM security requires this function to be highly compliant. If by any chance an IV is repeated for any or all instances of the authenticated encryption function which is having a key, then the entire implementation turns vulnerable for forgery attacks.

While encrypting the packet with GCM, the source card selects a key which is like the CBC algorithm, but the IV is selected based on a mechanism that ensures that the selected IV is unique and has never been used before, for any particular key. The AAD is included in the crypto header and once the encryption is complete, the authentication tag 'T' is added to the encrypted data before it is transmitted with the payload.

During the decryption process, the packet using GCM, the key and IV are selected using a similar mechanism like in the encryption process and the authentication tag is removed from the encrypted data. The AAD, the key and the IV are all used to decrypt the payload. If the authentication tag generated after decryption, matches with the authentication tag received from the source, then the integrity of data is ensured, and the decryption process is successful.

Encryption Method (iftask_aes_gcm_encrypt)

The new encryption method is given below:

- Determine the source card's slot number.
- Select the key and IV for this slot from the stored values.
- Generate a random number.
- Generate the *hash_index* for selecting from the dynamic IV table using source IP address, destination IP address and the random number.
- Generate the final IV by XORing the source cards IV with the IV from the dynamic IV table and then ORing it with the random number. This ensures that the final IV is unique for the key and the same key, IV pair is never re-used.



Note The size of the dynamic IV table is 64 and the random number is *uint16_t*.

- Select the IP fragment offset value as additional authentication data.
- Encrypt using the selected or generated key, IV and AAD.
- Fill the crypto header with the generated random number.

Decryption Method (iftask_aes_gcm_decrypt)

The new decryption method is given below:

- Determine the source card's slot number.
- From the stored values, select the key and IV for the source slot.
- Get the random number from the crypto header.
- Create the *hash_index* to select from the dynamic IV table, using the source IP address, destination IP address and the random number.
- Generate the final IV by XORing the IV of the source cards with the IV from the dynamic IV table and ORing it with the random number.
- Choose the IP fragment offset value as additional authentication data.
- Use AAD to authenticate the received payload.
- Proceed with encryption using the selected or generated key, IV and AAD.

Limitations

The following are the known limitations and restrictions of this feature:

- This feature is limited to the CUPS-DI systems only and not all VPC-DI systems.
- The change in encryption algorithm requires a reload. The algorithm can be modified by either manually modifying the */boot1/param.cfg* file or by using the new CLI to change the algorithm, before reloading and then initiating a reload.
- Only reloading with the preferred algorithm in the boot config without performing any changes before the reload will not lead to any change in the algorithm, as the encryption algorithm needs to be set before the card boot up process.
- The impact of the authentication algorithm on performance must be assessed due to any computational overheads of **aes-gcm-256** for smaller packets.

Configuring Encryption Algorithm

The encryption algorithm is configured through the boot parameter file, during the card boot up process. A new boot flag value *DI_NET_ENC_ALG* is available in the */boot1/param.cfg* file as the boot option during configuration.

The flag can be set using the CLI below or by manually editing the /boot1/param.cfg file. If it is set manually, it must be set to the same value in every CF and SF card for active and standby. 0 is for CBC, by default and 1 is for GCM.

Note For the changes to take effect, the CP has to be reloaded every time the encryption algorithm is changed.

Use the following configuration to configure the encryption algorithm in CUPS:

```
configure
   iftask di-net-encrypt-alg di_net_encrypt_alg
end
```

NOTES:

- **di-net-encrypt-alg**: Configures the encryption algorithm for the Di-LAN traffic. It represents the encryption algorithm name.

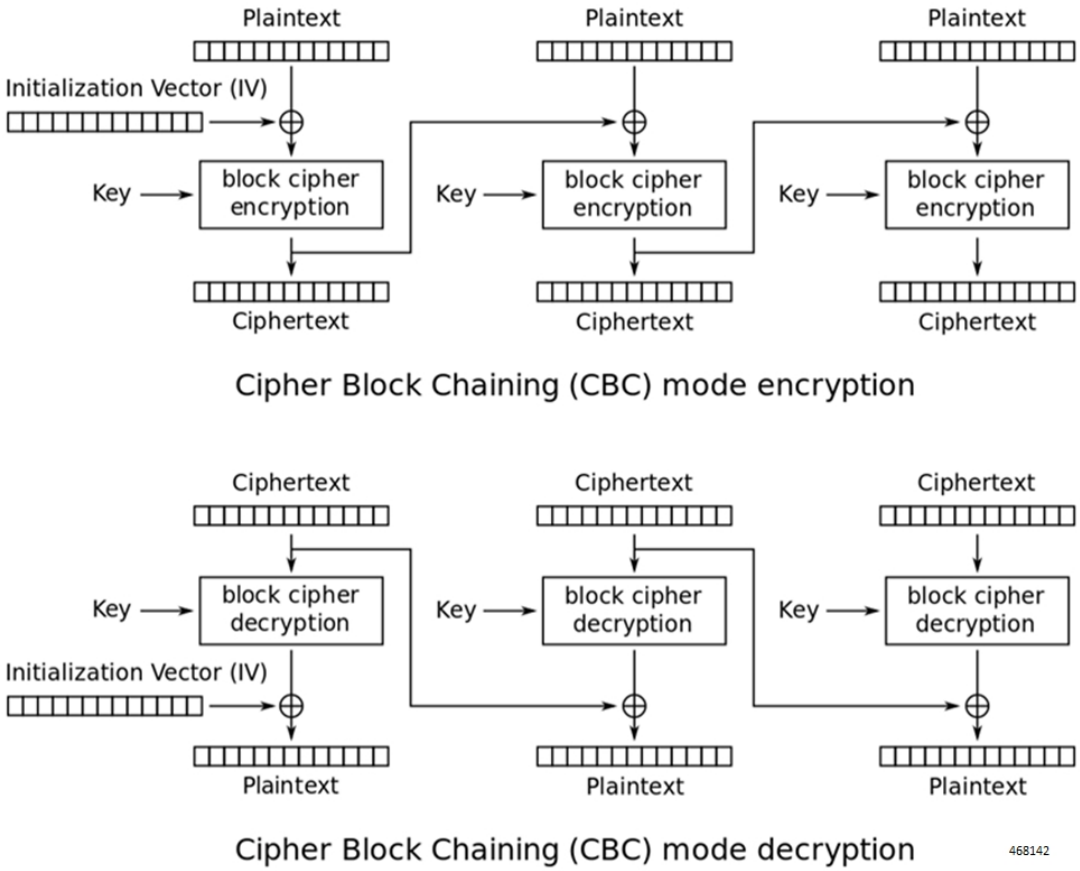
Appendix

Cipher Block Chaining

When plain text blocks are combined with cipher text blocks during encryption, in a confidential mode it is referred to as CBC. The CBC requires an unpredictable IV, which does not have to be a secret always to combine with the first plain text block.

Each plain text block is XORed with the previous cipher text block, making each cipher text block dependent on the plain text block before encryption, at any given point in time. To be unique each message IV must be used in the first block.

Figure 1: Cipher Block Chaining Method



Galois or Counter Mode

GCM combines the counter mode of encryption with the new Galois authentication mode. The key feature is the ease with which the Galois field multiplication used for authentication can be parallelized.

The two functions that comprise GCM are called authenticated encryption and decryption. The authenticated encryption function encrypts the confidential data and computes an authentication tag on both the confidential data and any additional, non-confidential data. The authenticated decryption function decrypts the confidential data, contingent on the verification of the tag.

Once a block cipher and key are selected and approved, the encryption function accepts the three input strings given below:

- Plain text, denoted as P.
- Additional Authenticated Data (AAD).
- Initialization Vector (IV).

GCM protects two types of data, the plain text and the AAD, by ensuring their authenticity. It also protects the confidentiality of the plain text while leaving the AAD transparent. The IV is a unique value that calls the authenticated encryption function on the input data that is to be protected.

The input string's bit length in the encryption algorithm must be within the limits given below:

- Length of P ≤ 239-256
- Length of A ≤ 264-1
- 1 ≤ Length of IV ≤ 264-1

The inputs for the authenticated encryption function are IV, AAD, secret key and plain text and the output is the cipher text having the same bit length as that of the plain text with the authentication tag T.

After approving and selecting a block cipher, key, and associated tag length, the IV, additional authenticated data A, cipher text C and authentication tag T are fed as inputs to the authenticated decryption function. The decryption process produced the outputs as follows:

- The plaintext P corresponding to the cipher text C.
- A special error code.



Note The output P indicates whether or not the authentication tag T for IV, A, and C was successful, otherwise the decryption process is considered as failed.

Figure 2: Galois or Counter Mode Method

