

Home

PUBLIC

Stack Overflow

Tags

Users

FIND A JOB

Jobs

Companies

TEAMS

What's this?

Free 30 Day Trial

How do you encrypt large files / byte streams in Go?

Asked 2 years, 8 months ago Active 2 years, 1 month ago Viewed 4k times

I have some large files I would like to AES encrypt before sending over the wire or saving to disk. While it seems possible to [encrypt streams](#), there seems to be [warnings](#) against [doing this](#) and instead people recommend splitting the files into chunks and using GCM or crypto/nacl/secretbox.

Processing streams of data is more difficult due to the authenticity requirement. We can't encrypt-then-MAC: by it's nature, we usually don't know the size of a stream. We can't send the MAC after the stream is complete, as that usually is indicated by the stream being closed. We can't decrypt a stream on the fly, because we have to see the entire ciphertext in order to check the MAC. Attempting to secure a stream adds enormous complexity to the problem, with no good answers. The solution is to break the stream into discrete chunks, and treat them as messages.

- <https://leanpub.com/gocrypto/read>

Files are segmented into 4KiB blocks. Each block gets a fresh random 128 bit IV each time it is modified. A 128-bit authentication tag (GHASH) protects each block from modifications.

- https://nuetzlich.net/gocryptfs/forward_mode_crypto/

If a large amount of data is decrypted it is not always possible to buffer all decrypted data until the authentication tag is verified. Splitting the data into small chunks fixes the problem of deferred authentication checks but introduces a new one. The chunks can be reordered... ..because every chunk is encrypted separately. Therefore the order of the chunks must be encoded somehow into the chunks itself to be able to detect rearranging any number of chunks.

- <https://github.com/minio/sio>

Can anyone with actual cryptography experience point me in the right direction?

Update

I realized after asking this question that there is a difference between simply not being able to fit the whole byte stream into memory (encrypting a 10GB file) and the byte stream *also* being an unknown length that could continue long past the need for the stream's start to be decoded (an 24-hour live video stream).

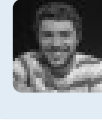
I am mostly interested in large blobs where the end of the stream can be reached before the beginning needs to be decoded. In other words, **encryption that does not require the whole plaintext/ciphertext to be loaded into memory at the same time.**

go encryption aes large-data

share improve this question follow

edited Mar 29 '18 at 23:13

asked Mar 29 '18 at 1:16

 **Xeoncross**
49.5k 72 236 344

are your files of well known size before you start the encryption process? or is their size calculated during the encryption window? – Ahmed Masud Mar 29 '18 at 1:20

Both, I have files on the filesystem / *know the size of*, and streams coming from a client that I don't trust. However, I impose a maximum byte size either way. – Xeoncross Mar 29 '18 at 1:30

add a comment

2 Answers

Active Oldest Votes

11

As you've already discovered from your research, there isn't much of an elegant solution for *authenticated* encryption of large files.

There are traditionally two ways to approach this problem:

- Split the file into chunks, encrypt each chunk individually and let each chunk have its own authentication tag. AES-GCM would be the best mode to use for this. This method causes file size bloating proportionate to the size of the file. You'll also need a unique nonce for each chunk. You also need a way to indicate where chunks begin/end.
- Encrypt using AES-CTR with a buffer, call `Hash.Write` on an HMAC for each buffer of encrypted data. The benefit of this is that encrypting can be done in one pass. The downside is that decryption requires one pass to validate the HMAC and then another pass to actually decrypt. The upside here is that the file size remains the same, plus roughly ~48 or so bytes for the IV and HMAC result.

Neither is ideal, but for very large files (~2GB or more), the second option is probably preferred.

I have included an example of encryption in Go using the second method below. In this scenario, the last 48 bytes are the IV (16 bytes) and the result of the HMAC (32 bytes). Note the HMACing of the IV also.

```
const BUFFER_SIZE int = 4096
const IV_SIZE int = 16

func encrypt(filePathIn, filePathOut string, keyAes, keyHmac []byte) error {
    inFile, err := os.Open(filePathIn)
    if err != nil { return err }
    defer inFile.Close()

    outFile, err := os.Create(filePathOut)
    if err != nil { return err }
    defer outFile.Close()

    iv := make([]byte, IV_SIZE)
    _, err = rand.Read(iv)
    if err != nil { return err }

    aes, err := aes.NewCipher(keyAes)
    if err != nil { return err }

    ctr := cipher.NewCTR(aes, iv)
    hmac := hmac.New(sha256.New, keyHmac)

    buf := make([]byte, BUFFER_SIZE)
    for {
        n, err := inFile.Read(buf)
        if err != nil && err != io.EOF { return err }

        outBuf := make([]byte, n)
        ctr.XORKeyStream(outBuf, buf[:n])
        hmac.Write(outBuf)
        outFile.Write(outBuf)


        if err == io.EOF { break }
    }

    outFile.Write(iv)
```

share improve this answer follow

edited Mar 29 '18 at 6:52

answered Mar 29 '18 at 1:48

 **Luke Joshua Park**
8,433 5 22 41

I modified [your code to run a real demo](#). How does AES-CTR compare to other schemes? How do you handle someone changing the order of blocks? **D.A.R.E** – Xeoncross Mar 29 '18 at 14:14

CTR is as close as you get to GCM, just without the authentication part. It's a stream cipher, which makes it perfect for this scenario. The only downside is that it falls apart if you reuse a nonce and key for two messages. It's easy to avoid doing this, however, by just always using a random nonce. – Luke Joshua Park Mar 29 '18 at 18:53

Can you clarify what you mean by "order of the blocks"? Do you mean for the first method I outlined? The second has no concept of blocks. The first would require some mechanism to account for ordering. yes. – Luke Joshua Park Mar 29 '18 at 18:54

4 Google drive client (in Go) uses **AES-CTR + AES-S12 HMAC** just as you show. – Xeoncross Mar 30 '18 at 16:15

3 GoCryptfs uses **GCM with 4kB ordered chunks** – Xeoncross Mar 30 '18 at 16:19

show 8 more comments

4

Using HMAC after encryption is a valid method. However, HMAC can be pretty slow, especially if SHA-2 is used. You could actually do the same with GMAC, the underlying MAC of GCM. It may be tricky to find an implementation but GMAC is over the ciphertext, so you can simply perform it separately if you really want. There are other methods as well such as Poly1305 with AES as used for TLS 1.2 and 1.3.

For GCM (or CCM or EAX or any other authenticated cipher) you need to authenticate the order of the chunks. You could do this by creating a separate file encryption key and then using the nonce input (the 12 byte IV) to indicate the number of the chunk. This will solve the storage of the IV *and* make sure that the chunks are in order. You can generate the file encryption key using a KDF (if you have a unique way to indicate the file) or by wrapping a random key with a master key.










share improve this answer follow

answered Mar 29 '18 at 15:48


 **Maarten Bodewes**
77.4k 12 116 217


add a comment


Your Answer

B I         

Sign up or [log in](#)

 Sign up using Google

 Sign up using Facebook

 Sign up using Email and Password

Post Your Answer

By clicking "Post Your Answer", you agree to our [terms of service](#), [privacy policy](#) and [cookie policy](#)

Post as a guest


Name


Email

Required, but never shown


Not the answer you're looking for? Browse other questions tagged [go](#) [encryption](#) [aes](#) [large-data](#) or [ask your own question](#).


The Overflow Blog


 How to write an effective developer resume: Advice from a hiring manager

 Podcast 290: This computer science degree is brought to you by Big Tech

Featured on Meta

 "Question closed" notifications experiment results and graduation

 MAINTENANCE WARNING: Possible downtime early morning Dec 2/4/9 UTC (8:30PM...)

 Congratulations VonC for reaching a million reputation

























Linked

- 3 Making GCM/CBC ciphers streamable in golang

Related

- 500 How to choose an AES encryption mode (CBC ECB CTR OCB CFB)?
- 4 AES CBC encryption of streams in ruby?
- 517 How can I convert a zero-terminated byte array to string?
- 7 Best approach to encrypt big files with php
- 1 Encrypting a Large File of irregular size using AES algorithm
- 0 Some chunks during RSA encryption on a split file result in too short ciphertext chunks
- 3 encrypting and decryption large file using rsa in java
- 3 Decrypting Large files with RSA in pycrypto?
- 1 Encrypting Files with AES C# in Chunks

Hot Network Questions

-  Density of States of Supercells
-  Nurimasaki Puzzle: Introduction
-  Example of X and Z are correlated, Y and Z are correlated, but X and Y are independent
-  Is Descriptive Complexity dead?
-  How can a hard drive provide a host device with file/directory listings when the drive isn't spinning?
-  Concatenating 's/' in raku
-  A colorful little Connect Wall
-  BJT and signal source
-  What happens if my Zurich public transportation ticket expires while I am traveling?
-  Prison planet book where the protagonist is given a quota to commit one murder a week
-  How easy it is to actually track another person credit card?
-  Have any other US presidents used that tiny table?
-  When and why did the use of the lifespans of royalty to limit clauses in contracts come about?
-  Tower of Pisa function
-  Best way to let people know you aren't dead, just taking pictures?
-  Did medieval people wear collars with a castellated hem?
-  Do I have to say Yes to "have you ever used any other name?" if I did?
-  Should live sessions be recorded for students when teaching a math course online?
-  Safe repair for electric shower casing
-  After what time interval do the closest approaches of Mercury to the Earth repeat?
-  Do it while you can or "Strike while the iron is hot" in French
-  How to say "garlic", "garlic clove" and "garlic bulb" in Japanese?
-  Why is "threepenny" pronounced as THREPNI?
-  How to calculate maximum input power on a speaker?

Question feed






PRODUCTS

Teams
Talent
Advertising
Enterprise

COMPANY

About
Press
Work Here
Legal
Privacy Policy
Contact Us

STACK EXCHANGE NETWORK

Technology 
Life / Arts 
Culture / Recreation 
Science 
Other 

Blog Facebook Twitter LinkedIn Instagram