Could not parse base64 DER-encoded ASN.1 public key from iOS in Golang

Ask Question

Asked 2 years, 9 months ago Active 2 years, 9 months ago Viewed 1k times

1

i have a projects in Golang with RSA enryption, so now, i have a Base64 public key format which used for encrypt a message,

i used this code:

```
publicKeyBase64 = "MIGJAoGBAJJYXgBem1scLKPEjwKrW8+ci3B/YNN3aY2DJ3lc5e2wNc0SmFikDpow1TdYcKl2
publicKeyBinary, err := base64.StdEncoding.DecodeString(publicKeyBase64)

publicKeyInterface, err := x509.ParsePKIXPublicKey(publicKeyBinary)
    if err != nil {
    fmt.Println("Could not parse DER encoded public key (encryption key)")
    return "","",err
}

publicKey, isRSAPublicKey := publicKeyInterface.(*rsa.PublicKey)
if !isRSAPublicKey {
    fmt.Println("Public key parsed is not an RSA public key")
    return "","",err
}

encryptedMessage, err := rsa.EncryptPKCS1v15(rand.Reader, publicKey, "message")
```

When i run this code, i got this error:

```
Could not parse DER encoded public key (encryption key)

asn1: structure error: tags don't match (16 vs {class:0 tag:2 length:129 isCompound:false})
```

The error points to publicKeyInterface, it failed to parse from Base64 decoded format to public Key, What's the problem with my code ?

=======updated=====

my publicKeyBase64 is retrieved from my models with Binary Data type

When i store it in my mongoDB from my Rails API, i receive public_key params as Base64 format, but i decode it to binary and then i stored it with this code

```
def create
  params = device_params
  public_key = Base64.decode64 device_params[:public_key]
  #device_params[:public_key] value is "MIGJAoGBAJJYXgBem1scLKPEjwKrW8+ci3B/YNN3aY2DJ3lc5e2
  params[:public_key] = BSON::Binary.new(public_key, :generic)
  device = Device.find_or_create_by(id: device_params[:id])

  render_success device.update_attributes(params), device
end
```

When i use rails code to convert my Base64 public key string using this code, it succeeded:

```
rsa_public_key = OpenSSL::PKey::RSA.new(Base64.decode64(public_key))
```

in my iOS app, i use https://github.com/DigitalLeaves/AsymmetricCrypto to generate a public Key using this code:

```
AsymmetricCryptoManager.sharedInstance.createSecureKeyPair({ (success, error) -> Void in
   if success {
    print("RSA-1024 keypair successfully generated.")
    let publicKey = AsymmetricCryptoManager.sharedInstance.getPublicKeyData()?.base64Encode

    let url = ENV.BASE_URL + "devices"
    let headers = ["Authentication-Token": CurrentUser.getCurrentUser().token] as! HTTPHead
    let params = ["device[user_id]": CurrentUser.getCurrentUser().id!, "device[id]": instar

    Alamofire.request(url, method: .post, parameters: params, encoding: URLEncoding.default
} else { print("An error happened while generating a keypair: \(error)") }
})
```

ios go swift3 rsa public-key-encryption

share improve this question follow edited Feb 13 '18 at 10:37

asked Feb 13 '18 at 6:50
calvin sugianto
440 ●4 ●20

- Can you show the code that generates the base64 key? – Marc Feb 13 '18 at 7:10
- I retrieve it from my models with Binary Data data type actually `publicKeyBase64 := base64.StdEncoding.EncodeToString(publicKey.Data)` – calvin sugianto Feb 13 '18 at 7:17
- What is `Data` ? That's not a field of a `rsa.PublicKey` . – Marc Feb 13 '18 at 7:22
- it's from my struct `PublicKey         bson.Binary` – calvin sugianto Feb 13 '18 at 7:23
- Ok, that still doesn't help. Where does it originally come from? Is it the `block.Bytes` from a pem block? If not, you're probably storing the wrong thing, or trying to decode it the wrong way. RIght now, we just know it's base64. – Marc Feb 13 '18 at 7:25

show 7 more comments

1 Answer

Active Oldest Votes

1

We can dump the ASN.1 contents to see what they look like:

```
$ echo "MIGJAoGBAJJYXgBem1scLKPEjwKrW8+ci3B/YNN3aY2DJ3lc5e2wNc0SmFikDpow1TdYcKl2wdrXX7sMRsy
    base64 -d | \
    dumpasn1 -
  0 137: SEQUENCE {
  3 129:   INTEGER
       :     00 92 58 5E 00 5E 9B 5B 1C 2C A3 C4 8F 02 AB 5B
       :     CF 9C 8B 70 7F 60 D3 77 69 8D 83 27 79 5C E5 ED
       :     B0 35 CD 12 98 58 A4 0E 9A 30 D5 37 58 70 A9 76
       :     C1 DA D7 5F BB 0C 46 CC A3 4E 4D 79 20 40 8C 7B
       :     3C 87 CD A2 46 43 D4 E2 9E 79 10 8B 12 7E 62 79
       :     4D 74 B0 B4 E1 33 56 3A 0D 77 A6 5C 9A 84 85 C1
       :     7E 8A D8 02 36 25 DB 05 48 03 C6 26 6B 66 C4 40
       :     58 66 00 59 04 5F 50 3F 43 57 48 2B 2E 84 7D 0F
       :     B1
135   3:   INTEGER 65537
       :   }

0 warnings, 0 errors.
```

A well-formatted ASN.1 public key should include the algorithm as well. We should have a line similar to:

```
  5   9:     OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
```

The `AsymmetricCryptoManager.getPublicKeyData()` returns a very barebones `ASN.1` key, without any algorithm information. This makes Go very unhappy as it has no way of knowing what kind of key it is. See more about correctly exporting the key here.

If you can change the iOS code, you should instead use CryptoExportImportManager and use one of `exportPublicKeyToPEM` or `exportPublicKeyToDER`. These take the output of `getPublicKeyData` and generate output usable by other tools. You can find an example of how to use them in the CryptoExportImportManager example.

If you cannot change the key export code, you can instead parse it directly in Go. This assumes that you know for sure that it is a RSA public key:

```
func main() {
    publicKeyBase64 := "MIGJAoGBAJJYXgBem1scLKPEjwKrW8+ci3B/YNN3aY2DJ3lc5e2wNc0SmFikDpow1Tc

    // Base64 decode.
    publicKeyBinary, err := base64.StdEncoding.DecodeString(publicKeyBase64)
    if err != nil {
        panic(err)
    }

    // rsa.PublicKey is a big.Int (N: modulus) and an integer (E: exponent).
    var pubKey rsa.PublicKey
    if rest, err := asn1.Unmarshal(publicKeyBinary, &pubKey); err != nil {
        panic(err)
    } else if len(rest) != 0 {
        panic("rest is not nil")
    }

    fmt.Printf("key: %+v\n", pubKey)
}
```

This prints out:

> key: {N:+102767083290202280873554060983826675083148443795791447833515664566475334389364583758312108980110921996262487865832851258326049062353432991986398760705560379825908169063986770245967781444794847106351934016144540466696422397564949226710181429429140226472206572796987719088983654589217713611861345869296293449649 E:65537}

You can now use your public key in package rsa functions.

share improve this answer follow edited Feb 13 '18 at 10:51

answered Feb 13 '18 at 9:17
Marc
13k ●5 ●29 ●36

- I use another Base64 encoded from my friend (comes from android, but don't know yet what library he's using right now) and it works fine with current code `"MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDkx9h4KXOoW+2Kb06pLz7vo3GN3iGqWk8sS2SxvRSM+/S6bmhrt2Mw+StSSzQSn4YMhF1rEU7fED1buI85bx8QpFwdAu1MzB1uWVXSo/koM3voAzghQTiCvoJigxgIHtKdr2e03D9vWYQPxh862yFIstzItCgksliSwLXD5++IKQIDAQAB"` – calvin sugianto Feb 13 '18 at 9:46
- I think it is better to change my iOS code to send valid public Key with Base64 encoded format – calvin sugianto Feb 13 '18 at 9:47
- Your friend's key includes the algorithm: `echo ... | base64 -d | dumpasn1 -` shows `.... OBJECT IDENTIFIER rsaEncryption ...`. Yours did not, just integers. – Marc Feb 13 '18 at 9:57
- And I agree, if you can change the iOS code, that will be much more durable. Other libraries/tools will have the same problem as Go. It's also good if you want to allow EC keys. – Marc Feb 13 '18 at 10:01
- i think, i prefer to fix the `AsymmetricCryptoManager.getPublicKeyData()` in `AsymmetricCryptoManager.swift` to add the algorithm information, but i'm still confused how to add it. I have change the default of `kAsymmetricCryptoManagerKeySize` value to 1024 from 2048 (because i just need 1024) – calvin sugianto Feb 13 '18 at 10:36

show 12 more comments

Your Answer

Sign up or log in

Sign up using Google

Sign up using Facebook

Sign up using Email and Password

Post as a guest

Name

Email

Required, but never shown

Post Your Answer

By clicking "Post Your Answer", you agree to our terms of service, privacy policy and cookie policy

Not the answer you're looking for? Browse other questions tagged ios go swift3 rsa public-key-encryption or ask your own question.

Related

- 10 RSA encrypt with base64 encoded public key in Android
- 1 Convert XML Dsig format to DER ASN.1 public key
- 145 RSA Public Key format
- 0 Convert RSA Public key (2048 bit) from XML format to DER ASN.1 public key for iOS
- 16 Creating RSA Public Key From String
- 2 How to Decode RSA public key(in java) from a text view in Android studio
- 0 Android RSA Encryption throws InvalidKeySpecException

Hot Network Questions

- "Hello, World!" in zero lines of code
- How does the title "Revenge of the Sith" suit the plot?
- Should live sessions be recorded for students when teaching a math course online?
- How can a hard drive provide a host device with file/directory listings when the drive isn't spinning?
- Example of X and Z are correlated, Y and Z are correlated, but X and Y are independent
- Choose Snapping Point?
- Prevent sagging shelves in alcove
- Have any other US presidents used that tiny table?
- What is the timeline for using arXiv in computer science?
- Why was the name of Discovery's most recent episode "Unification III"?
- What does "中" as an independent verb means?
- How to calculate current flowing through this diode?
- Can you buy a property on your next roll?
- How many pawns make up for a missing queen in the endgame?
- How do I legally resign in Germany when no one is at the office?
- How much do propellers stutter?
- Merging values into an object
- Construct a polyhedron from the coordinates of its vertices and calculate the area of each face
- After what time interval do the closest approaches of Mercury to the Earth repeat?
- Is it considered offensive to address one's seniors by name in the US?
- Why is SQL Server's STDistance Very Slightly Different Than The Vincenty Formula? (Same Up To ~0.0001km)
- Joining Tikz paths seamlessly
- Are Van Der Waals Forces the Similar to Van der Waal Equation?
- How to backfill trench under slab in Los Angeles

Question feed