RSA και Ψηφιακές Υπογραφές

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 1

Οικονόμου Βασίλειος | Τμήμα: ΑΣΦ01

Σημείωση

Δημιουργήθηκε script για την εργασία που δέχεται σαν παράμετρο προαιρετικά ένα string για την δραστηριότητα 2. Το αρχείο `exercise_1.sh` παράγει τα αποτελέσματα για όλες τις δραστηριότητες. Λειτουργία του είναι να μεταγλωττίζει το `exercise_1.c` και έπειτα τρέχει το εκτελέσιμο με ορίσματα ένα την παράμετρο του script και τα περιεχόμενα των αρχείων. Επίσης, περιέχονται σε σχόλια οι εντολές που έτρεξαν για την δραστηριότητα 6.

Δραστηριότητα 1: Δημιουργία ιδιωτικού κλειδιού

Δεδομένα:

- P1 = 953AAB9B3F23ED593FBDC690CA10E703
- P2 = C34EFC7C4C2369164E953553CDF94945
- e = 0D88C3

Υπολογισμός του ιδιωτικού κλειδιού

Αρχικά δημιουργείται ο αριθμός n ως εξής:

$$n = \text{prime1} \times \text{prime2}$$

Υπολογίζουμε το $\Phi(n)$ με την παρακάτω σχέση:

$$\Phi(n) = \Phi(\text{Prime1}) \times \Phi(\text{prime2}) \Leftrightarrow \Phi(n) = (\text{prime1} - 1) \times (\text{Prime2} - 1)$$

Τέλος, το ιδιωτικό κλειδί υπολογίζεται με την σχέση:

$$d = e^{-1} \times \text{mod}(\Phi(n))$$

Η εξής διαδικασία μεταφράζεται σε κώδικα C:

```
BN_mul(n, p1, p2, ctx);
BN_sub(phi_p1, p1, BN_value_one());
BN_sub(phi_p2, p2, BN_value_one());
BN_mul(phi_n, phi_p1, phi_p2, ctx);
BN_mod_inverse(d, e, phi_n, ctx);
```

Η εκτέλεση του κώδικα με επιπρόσθετες εντολές εμφάνισης παράγουν στο τερματικό την έξοδο:

```
p1:  953AAB9B3F23ED593FBDC690CA10E703
p2:  C34EFC7C4C2369164E953553CDF94945
e:  0D88C3
n:  71D9BBC5C01F9B50DDFE5F2EC331FAB21081009D014E9615C277670C61591ECF
private key: 63F67E805D8DEB0B4182C57C3DC24F3C1350CF182E8ABF85FD24062A3BC7F2EB
```

Δραστηριότητα 2: Κρυπτογράφηση μηνύματος

Δεδομένα:

- n: 71D9BBC5C01F9B50DDFE5F2EC331FAB21081009D014E9615C277670C61591ECF
- e = 0D88C3
- d = 63F67E805D8DEB0B4182C57C3DC24F3C1350CF182E8ABF85FD24062A3BC7F2EB
- argv[1] = Oikonomoy Vasileios

Κρυπτογράφηση μηνύματος

Στην ανάπτυξη του προγράμματος υλοποιήθηκαν σε C συναρτήσεις για την κωδικοποίηση και την αποκωδικοποίηση των χαρακτήρων σε δεκαεξαδικό και αντίστροφα. Άρα η μέθοδος `encode` και `decode` της python αντικαταστάθηκε με την συνάρτηση `str2hex` και `hex2str`.

Το μήνυμα είναι η πρώτη παράμετρος του προγράμματος και γίνεται η χρήση της argv[1]

Μετά την κωδικοποίηση του μηνύματος γίνεται η κρυπτογράφηση. Σε αυτό το στάδιο χρησιμοποιείται το public key(e) με την εξής σχέση:

$$\text{encrypt\_message} = \text{message}^{e} \times \text{mod}(n)$$

Και η αποκρυπτογράφηση γίνεται αντίστοιχα:

$$\text{decrypt\_message} = \text{encrypt\_message}^{d} \times \text{mod}(n)$$

Η εξής διαδικασία μεταφράζεται σε κώδικα C:

```c
/* Encryption */
BN_mod_exp(encrypt_message, message, e, n, ctx);
/* Decryption */
BN_mod_exp(decrypt_message, encrypt_message, d, n, ctx);
```

Η εκτέλεση του κώδικα με επιπρόσθετες εντολές εμφάνισης παράγουν στο τερματικό την έξοδο:

message: "Oikonomoy Vasileios" convert to:"4F696B6F6E6F6D6F7920566173696C65696F73"

Encrypted message: 5FC14EBEE5ADD8730C026E6421E840FAA682966A76641BF47783D7BB63BAE90A

Decrypted message: 4F696B6F6E6F6D6F7920566173696C65696F73

Message: Oikonomoy Vasileios

Δραστηριότητα 3: Αποκρυπτογράφηση μηνύματος

Δεδομένα:

- n: DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5
- e = 010001
- d = 74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D
- C = CAF7D72776AFEFBAC8269E1A8B76CE44A3B28015CA9A54E22C239EF38FCFAFFA

Αποκρυπτογράφηση μηνύματος

Για την αποκρυπτογράφηση του μηνύματος ακολουθείται η ίδια διαδικασία

$$\text{decrypt\_message} = c^{d} \times \text{mod}(n)$$

Η εξής διαδικασία μεταφράζεται σε κώδικα C:

```
/* Decryption */
BN_mod_exp(decrypt_message, c, d, n, ctx);

/* Display message in hex and ascii form */
printBN("Decrypted message: ", decrypt_message);
DispBN2str(decrypt_message);
```

Η συνάρτηση DispBN2str αποκωδικοποιεί το μήνυμα και το τυπώνει στην οθόνη.

Η εκτέλεση του κώδικα παράγουν στο τερματικό την έξοδο:

C Encrypted message:
CAF7D72776AFEFBAC8269E1A8B76CE44A3B28015CA9A54E22C239EF38FCFAFFA

Decrypted message: 494E464F53454320537072696E672032303232

Message: INFOSEC Spring 2022

Δραστηριότητα 4: Υπογραφή μηνύματος

Δεδομένα:

- n: DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5
- e = 010001
- d = 74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D
- message = Oikonomoy Vasileios

Υπογραφή

Για την υπογραφή ενός μηνύματος αντιστρέφονται οι ρόλοι τον κλειδιών. Κρυπτογραφούμε με το ιδιωτικό κλειδί, δηλαδή με το κλειδί αποκρυπτογράφησης όπως φαίνεται παρακάτω.

$$\text{signature} = \text{message}^{d} \times \text{mod}(n)$$

Η εξής διαδικασία μεταφράζεται σε κώδικα C:

```
/* Encryption with private key(variable: d) */
BN_mod_exp(signature_1, message, d, n, ctx);

/* Change message */
hex_string = str2hex("Oikonomou Vasileios");
if(hex_string == NULL)
	return 0;

/* Encode the message to hex */
BN_hex2bn(&alter_message, hex_string);

/* Encryption with private key(variable: d) */
BN_mod_exp(signature_2, alter_message, d, n, ctx);

/* Compare the two signs */
printf("\nsign_1 and sign_2 are %sequal \n", BN_cmp(signature_1, signature_2)?
"NOT " : "");
```

Η εκτέλεση του κώδικα παράγουν στο τερματικό την έξοδο:

```
FirstMessage: Oikonomoy Vasileios

signed message:
D3BBC32862C94369F1253D3D05C4FF74AC560528381520D2AAE4E88E74FA001D
```

```
Second Message: Oikonomou Vasileios

signed message:  453485B920902BCCA776182D685442D8413D5DA4FFF5E770F6E1629AB8A8757B


sign_1 and sign_2 are NOT equal
```

Δραστηριότητα 5: Επαλήθευση Υπογραφής

Δεδομένα:

- n: DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5
- e = 010001
- message = Launch a missile.

Επαλήθευση Υπογραφής

Για την επαλήθευση της υπογραφής, με δεδομένου ότι η υπογραφή δημιουργήθηκε με το private key, χρησιμοποιώντας το public key στην υπογραφή θα επιστρέψει το αρχικό μήνυμα.

$$\text{message} = \text{signature}^{e} \times \text{mod}(n)$$

Η εξής διαδικασία μεταφράζεται σε κώδικα C:

```c
/* Encode the message to hex */
BN_hex2bn(&message, hex_string);    //Original message to variable: message

/* Verification */
BN_mod_exp(verification, signature_1, e, n, ctx);

/* Check if produced the Message */
printf("Message and verification are %sequal\n\n", BN_cmp(message, verification)?
"NOT " : "");

/* Modify sign  */
BN_hex2bn(&signature_2, "643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542C
BDB6803F");

/* Verification on modified sign */
BN_mod_exp(verification, signature_2, e, n, ctx);
```

```
/* Check if produced the Message */
printf("\nsign_1 and sign_2 are %sequal\n\n", BN_cmp(signature_1, signature_2)?
"NOT " : "");

/* Verification */
BN_mod_exp(verification, signature_1, e, n, ctx);

/* Check if produced the Message */
printf("Message and verification are %sequal\n", BN_cmp(message, verification)?
"NOT " : "");
```

Η εκτέλεση του κώδικα παράγουν στο τερματικό την έξοδο:

```
Message: Launch a missile.

Message(hex): 4C61756E63682061206D697373696C652E

Message sign:
643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6802F

Message verification: 4C61756E63682061206D697373696C652E


Message and verification are equal


modified sign:
643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6803F

Modified message verification:
91471927C80DF1E42C154FB4638CE8BC726D3D66C83A4EB6B7BE0203B41AC294


sign_1 and sign_2 are NOT equal

        Case B

Message: Please transfer me $2000.Alice.

Message(hex): 506C65617365207472616E73666572206D652024323030302E416C6963652E

Message sign: DB3F7CDB93483FC1E70E4EACA650E3C6505A3E5F49EA6EDF3E95E9A7C6C7A320

Message verification: 506C65617365207472616E73666572206D652024323030302E416C6963652E


Message and verification are equal
```

Δραστηριότητα 6: Επαλήθευση πιστοποιητικού

Δεδομένα:

- Επιλέχτηκε η ιστοσελίδα: www.gsis.gr

Επαλήθευση πιστοποιητικού X.509

Εκτελείται η παρακάτω εντολή για την εξαγωγή των πιστοποιητικών

```
openssl s_client -connect www.gsis.gr:443 -showcerts
```

Από την έξοδο αντιγράφηκαν τα πιστοποιητικά στα αρχεία c0.pem και c1.pem.

C0.pem

-----BEGIN CERTIFICATE-----
MIIGIzCCBQugAwIBAgIQAzIplc8rQVfjAmQH0+ZnFTANBgkqhkiG9w0BAQsFADBN
MQswCQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMScwJQYDVQQDEx5E
aWdpQ2VydCBTSEEyIFNlY3VyZSBTZXJ2ZXIgQ0EwHhcNMjEwNzE1MDAwMDAwWhcN
MjIwNzIwMjM1OTU5WjB5MQswCQYDVQQGEwJVUzEWMBQGA1UECBMNTWFzc2FjaHVz
ZXR0czESMBAGA1UEBxMJQ2FtYnJpZGdlMSIwIAYDVQQKExlBa2FtYWkgVGVjaG5v
bG9naWVzLCBJbmMuMRowGAYDVQQDExFhMjQ4LmUuYWthbWFpLm5ldDBZMBMGByqG
SM49AgEGCCqGSM49AwEHA0IABHUvlRoaw+ocKVl6w1xqKQ0zRNpdXdHNzf96RuTZ
1u0BJWVgzXZTx9z5dHIZUN8BIfLekZA1WAWJgwkwkiXpbyejggOcMIIDmDAfBgNV
HSMEGDAWgBQPgGEcgjFh1S8o541GOLQs4cbZ4jAdBgNVHQ4EFgQUZDZqoE0eKutf
AwhNuY78VXN3bTEwbgYDVR0RBGcwZYIRYTI0OC5lLmFrYW1haS5uZXSCDyouYWth
bWFpemVkLm5ldIIXKi5ha2FtYWl6ZWQtc3RhZ2luZy5uZXSCDiouYWthbWFpaGQu
bmV0ghYqLmFrYW1haWhkLXN0YWdpbmcubmV0MA4GA1UdDwEB/wQEAwIHgDAdBgNV
HSUEFjAUBggrBgEFBQcDAQYIKwYBBQUHAwIwawYDVR0fBGQwYjAvoC2gK4YpaHR0
cDovL2NybDMuZGlnaWNlcnQuY29tL3NzY2Etc2hhMi1nNy5jcmwwL6AtoCuGKWh0
dHA6Ly9jcmw0LmRpZ2ljZXJ0LmNvbS9zc2NhLXNoYTItZzcuY3JsMD4GA1UdIAQ3
MDUwMwYGZ4EMAQICMCkwJwYIKwYBBQUHAgEWG2h0dHA6Ly93d3cuZGlnaWNlcnQu
Y29tL0NQUzB8BggrBgEFBQcBAQRwMG4wJAYIKwYBBQUHMAGGGGh0dHA6Ly9vY3Nw
LmRpZ2ljZXJ0LmNvbTBGBggrBgEFBQcwAoY6aHR0cDovL2NhY2VydHMuZGlnaWNl
cnQuY29tL0RpZ2lDZXJ0U0hBMlNlY3VyZVNlcnZlckNBLmNydDAMBgNVHRMBAf8E
AjAAMIIBfAYKKwYBBAHWeQIEAgSCAWwEggFoAWYAdQApeb7wnjk5IfBWc59jpXfl
vld9nGAK+PlNXSZcJV3HhAAAAXqqC4h/AAAEAwBGMEQCIBU9iuh6L2j216LPLezY
RusSstNnDapGUX6OsDUgA5uKAiApoOykO5K7178p4pyrAEEwl6z1teXyedhfx7v8
8v1cNgB1AFGjsPX9AXmcVm24N3iPDKR6zBsny/eeiEKaDf7UiwXlAAABeqoLiH4A
AAQDAEYwRAIgblVlRgNXy/h973OfuDZ+eu2p/YcbyeZczpm1UQGb23kCIFFX46pV
gUNpyOm4yn+2eqgjBj7Uf0EZyVHhJt5puFTpAHYAQcjKsd8iRkoQxqE6CUKHXk4x
ixsD6+tLx2jwkGKWBvYAAAF6qguIFAAABAMARzBFAiEA5i3QE1vCNEGkZi0DQNp8
xIGPIIkwfxytougz4u33d6kCIFE17NC8x6WMqGYYGiRfw4BnSwWU1Pyi7DbNBOs6
9pRyMA0GCSqGSIb3DQEBCwUAA4IBAQAmUzvkyK/VObisTSu4WnZaZjPSNf7QKNNX
/wypkVWcf1es1H4dRGZ08PcuYKnLCLtN6RKL75+Rffb+c7XunfY7rOQ9ylgyVa6i
6NGKF42KYPpZ9wNbjX4fkshsaI95OVZkE8NqupnT6T8U30ubfpOGNGRstbkHawWV
FMk9NJCY9eoELu3lwibVCu39kMlP6gCNP6i5E0S3rimuFqWCoHD8seR7D8JNqOej
Ekg0KyLP8jm3LhqNibnZ3/f4YET1IfpkNG0PZcp87C6OpiQXSSku01Uq83GhuTcD
lnxlPA7grXnWprtyUhDPLejNjsx82yTn2vKcTKUzd99vC5SxWiMZ
-----END CERTIFICATE-----

C1.pem

```
-----BEGIN CERTIFICATE-----
MIIElDCCA3ygAwIBAgIQAf2j627KdciIQ4tyS8+8kTANBgkqhkiG9w0BAQsFADBh
MQswCQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMRkwFwYDVQQLExB3
d3cuZGlnaWNlcnQuY29tMSAwHgYDVQQDExdEaWdpQ2VydCBHbG9iYWwgUm9vdCBD
QTAeFw0xMzAzMDgxMjAwMDBaFw0yMzAzMDgxMjAwMDBaME0xCzAJBgNVBAYTAlVT
MRUwEwYDVQQKEwxEaWdpQ2VydCBJbmMxJzAlBgNVBAMTHkRpZ2lDZXJ0IFNIQTIg
U2VjdXJlIFNlcnZlciBDQTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
ANyuWJBNwcQwFZA1W248ghX1LFy949v/cUP6ZCWA1O4Yok3wZtAKc24RmDYXZK83
nf36QYSvx6+M/hpzTc8zl5CilodTgyu5pnVILR1WN3vaMTIa16yrBvSqXUu3R0bd
KpPDkC55gIDvEwRqFDu1m5K+wgdlTvza/P96rtxcflUxDOg5B6TXvi/TC2rSsd9f
/ld0Uzs1gN2ujkSYs58O09rg1/RrKatEp0tYhG2SS4HD2nOLEpdIkARFdRrdNzGX
kujNVA075ME/OV4uuPNcfhCOhkEAjUVmR7ChZc6gqikJTvOX6+guqw9ypzAO+sf0
/RR3w6RbKFfCs/mC/bdFWJsCAwEAAaOCAVowggFWMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDgYDVR0PAQH/BAQDAgGGMDQGCCsGAQUFBwEBBCgwJjAkBggrBgEFBQcwAYYY
aHR0cDovL29jc3AuZGlnaWNlcnQuY29tMHsGA1UdHwR0MHIwN6A1oDOGMWh0dHA6
Ly9jcmwzLmRpZ2ljZXJ0LmNvbS9EaWdpQ2VydEdsb2JhbFJvb3RDQS5jcmwwN6A1
oDOGMWh0dHA6Ly9jcmw0LmRpZ2ljZXJ0LmNvbS9EaWdpQ2VydEdsb2JhbFJvb3RD
QS5jcmwwPQYDVR0gBDYwNDAyBgRVHSAAMCowKAYIKwYBBQUHAgEWHGh0dHBzOi8v
d3d3LmRpZ2ljZXJ0LmNvbS9DUFMwHQYDVR0OBBYEFA+AYRyCMWHVLyjnjUY4tCzh
xtniMB8GA1UdIwQYMBaAFAPeUDVW0Uy7ZvCj4hsbw5eyPdFVMA0GCSqGSIb3DQEB
CwUAA4IBAQAjPt9L0jFCpbZ+QlwaRMxp0Wi0XUvgBCFsS+JtzLHgl4+mUwnNqipl
5TlPHoOlblyYoiQm5vuh7ZPHLgLGTUq/sELfeNqzqPlt/yGFUzZgTHbO7Djc1lGA
8MXW5dRNJ2Srm8c+cftIl7gzbckTB+6WohsYFfZcTEDts8Ls/3HB40f/1LkAtDdC
2iDJ6m6K7hQGrn2iWZiIqBtvLfTyyRRfJs8sjX7tN8Cp1Tm5gr8ZDOo0rwAhaPit
c+LJMto4JQtV05od8GiG7S5BNO98pVAdvzr508EIDObtHopYJeS4d60tbvVS3bR0
j6tJLp07kzQoH3jOlOrHvdPJbRzeXDLz
-----END CERTIFICATE-----
```

Με την παρακάτω εντολή τυπώνεται το modulo του c1 πιστοποιητικού και αποθηκεύεται χειροκίνητα στο αρχείο modulo.txt

```
openssl x509 -in c1.pem -noout -modulus
Modulus=DCAE58904DC1C4301590355B6E3C8215F52C5CBDE3DBFF7143FA642580D4EE18A24DF066D0
0A736E1198361764AF379DFDFA4184AFC7AF8CFE1A734DCF339790A2968753832BB9A675482D1D5637
7BDA31321AD7ACAB06F4AA5D4BB74746DD2A93C3902E798080EF13046A143BB59B92BEC207654EFCD
AFCFF7AAEDC5C7E55310CE83907A4D7BE2FD30B6AD2B1DF5FFE5774533B3580DDAE8E4498B39F0ED3
DAE0D7F46B29AB44A74B58846D924B81C3DA738B129748900445751ADD37319792E8CD540D3BE4C13
F395E2EB8F35C7E108E8641008D456647B0A165CEA0AA29094EF397EBE82EAB0F72A7300EFAC7F4FD14
77C3A45B2857C2B3F982FDB745589B
```

Με την έξοδο της παρακάτω εντολής εντοπίζεται το δημόσιο κλειδί του c1 πιστοποιητικό και αποθηκεύεται στο αρχείο public_key.txt

```
openssl x509 -in c1.pem -text -noout
```

Ομοίως και με την διαδικασία του δημόσιου κλειδιού, αλλά αυτή την φορά επιλέγεται η υπογραφή και αποθηκεύεται στο αρχείο signature.txt

```
openssl x509 -in c0.pem -text -noout
```

Η διαδικασία ανάκτησης του hash γίνεται με τρεις εντολές. Με την πρώτη εντολή τυπώνουμε όλη την δομή ώστε να αναγνωριστεί το κομμάτι που περιέχει το hash. Με την δεύτερη εντολή δίνεται ο περιορισμός ώστε να απομονωθεί και να αποθηκευτεί στο αρχείο c0_body.bin. Και τέλος, με την κρυπτογραφική συνάρτηση sha256 εξάγεται και αποθηκεύεται χειροκίνητα στο αρχείο hash.txt.

```
openssl asn1parse -i -in c0.pem

# #Certificate
openssl asn1parse -i -in c0.pem -strparse 4 -out c0_body.bin -noout

sha256sum c0_body.bin
```

Έχοντας τα παρακάτω αρχεία:

- public_key.txt
- modulo.txt
- signature.txt
- hash.txt

Ο κώδικας που εκτελέστηκε για την επαλήθευση του hash όπως και με τις προηγούμενες δραστηριότητες:

```
/* Verification */
    BN_mod_exp(verification, signature_1, e, n, ctx);
```

Η εκτέλεση του κώδικα με επιπρόσθετες εντολές εμφάνισης παράγουν στο τερματικό την έξοδο:

```
public key: 010001


modulo:
DCAE58904DC1C4301590355B6E3C8215F52C5CBDE3DBFF7143FA642580D4EE18A24DF066D00
A736E1198361764AF379DFDFA4184AFC7AF8CFE1A734DCF339790A2968753832BB9A675482D
1D56377BDA31321AD7ACAB06F4AA5D4BB74746DD2A93C3902E798080EF13046A143BB59B92
BEC207654EFCDAFCFF7AAEDC5C7E55310CE83907A4D7BE2FD30B6AD2B1DF5FFE5774533B358
0DDAE8E4498B39F0ED3DAE0D7F46B29AB44A74B58846D924B81C3DA738B129748900445751
ADD37319792E8CD540D3BE4C13F395E2EB8F35C7E108E8641008D456647B0A165CEA0AA2909
4EF397EBE82EAB0F72A7300EFAC7F4FD1477C3A45B2857C2B3F982FDB745589B


signature:
26533BE4C8AFD539B8AC4D2BB85A765A6633D235FED028D357FF0CA991559C7F57ACD47E1D4
46674F0F72E60A9CB08BB4DE9128BEF9F917DF6FE73B5EE9DF63BACE43DCA583255AEA2E8D18
A178D8A60FA59F7035B8D7E1F92C86C688F7939566413C36ABA99D3E93F14DF4B9B7E9386346
46CB5B9076B059514C93D349098F5EA042EEDE5C226D50AEDFD90C94FEA008D3FA8B91344B7
```

AE29AE16A582A070FCB1E47B0FC24DA8E7A31248342B22CFF239B72E1A8D89B9D9DFF7F8604
4F521FA64346D0F65CA7CEC2E8EA6241749292ED3552AF371A1B93703967C653C0EE0AD79D6
A6BB725210CF2DE8CD8ECC7CDB24E7DAF29C4CA53377DF6F0B94B15A2319

hash: D4938B01D79C2C02713B30B7E420D43C96819308947F0AD7DF01344334E5AFFD

Message verification:
01FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF003031300D0609608
648016503040201050000420D4938B01D79C2C02713B30B7E420D43C96819308947F0AD7DF013
44334E5AFFD