University of the Aegean, Samos, **2017-2018**
Information & Communication Systems Engineering - **ICSD**

Ασφάλεια Δικτύων Υπολογιστών & Τεχνολογίες Προστασίας της Ιδιωτικότητας

MultiKAP

(Διδάσκων)
Γιώργος Καμπουράκης

(Εργαστηριακοί Συνεργάτες)
Αλέξανδρος Φακής
Νίκος Αλεξκίου

Νίκος Τρίτσης
(icsd11162)

Χρήστος Αυλακιώτης
(icsd12015)

University of the Aegean, Samos, **2017-2018**
Information & Communication Systems Engineering - **ICSD**

Ασφάλεια Δικτύων Υπολογιστών & Τεχνολογίες Προστασίας της Ιδιωτικότητας

MultiKAP

(Διδάσκων)
Γιώργος Καμπουράκης

(Εργαστηριακοί Συνεργάτες)
Αλέξανδρος Φακής
Νίκος Αλεξκίου

Νίκος Τρίτσης
(icsd11162)

Χρήστος Αυλακιώτης
(icsd12015)

Περιεχόμενα

Εισαγωγή

Σε αυτή την εργασία ασχοληθήκαμε με την υλοποίηση τριών διαφορετικών πρωτοκόλων για τη συμφωνία ενός κοινού συμμετρικού κλειδιού για την ασφαλή επικοινωνία μεταξύ 2 οντοτήτων σε ένα μη-ασφαλές κανάλι.

Τα 3 αυτά πρωτόκολλα είναι τα:

- Encapsulation KAP
- Diffie-Hellman KAP
- Station-To-Station KAP

Στα πλαίσια της εργασίας έχει δημιουργηθεί ένα εύχρηστο γραφικό περιβάλλον που επιτρέπει στον χρήστη να επιλέξει ένα απ'τα υλοποιημένα πρωτόκολλα που θα χρησιμοποιηθεί σε ένα απλό session μεταξύ 2 οντοτήτων (στο παράδειγμα μας Alice & Bob), κατα το οποίο οι 2 οντότητες έρχονται σε συμφωνία ενός κοινού συμμετρικού κλειδιού που θα εξασφαλίσει την ασφαλή επικοινωνία τους στο μέλλον.

Security provider σε όλες τις υλοποιήσεις είναι ο BouncyCastleProvider.

Τα Certificates δημιουργούνται μέσω αιτήματος πιστοποίησης με χρήση της κλάσσης PKCS10CertificationRequest της Bouncy Castle.

Για τη CA δημιουργήσαμε ένα αρχείο keystore που σε αυτό βάλαμε το ανθυπόγραφο πιστοποιητικό και το private της και του χρησιμοποιήσαμε ως TrustStore (δηλώσαμε το KeyStore ως έμπιστο).

Τέλος, τα συμμετρικά κλειδιά που δημιουργούνται έχουν μήκος 256bit

Σημείωση:
Designed for Two-Page View & Show Cover Page (PDF reader settings)

1. Encapsulation Key Agreement Protocol

Περιγραφή Πρωτοκόλλου:

Το πρωτόκολλο ενθυλάκωσης περιλαμβάνει τα ακόλουθα βήματα:
(οι ακόλουθες ενέργειες εκτελούνται απ’τις 2 οντότητες - Alice & Bob)

Alice: αποστέλει στον Bob το δημόσιο κλειδί της.
Bob: παραλμβάνει το δημόσιο κλειδί της Alice.
Bob: δημιουργεί το κοινό συμμετρικό κλειδί που θα χρησιμοποιηθεί για τη μεταξύ τους επι κοινωνία.
Bob: ενθυλακώνει αυτό το κλειδί με το δημόσιο κλειδί της Alice και το στέλνει στην Alice.
Alice: παραλαμβάνει το κλειδί που απέστειλε ο Bob.
Alice: με το ιδιωτικό κλειδί της απενθυλακώνει το κλειδί που έστειλε ο Bob .

Alice: κρυπτογραφεί ένα μήνυμα με το συμμετρικό κλειδί που διαθέτει πλέον.
Bob: αποκρυπτογραφεί με τη σειρά του το μήνυμα με το κοινό (πλέον) συμμετρικό κλειδί.

Τεχνικές Λεπτομέρειες Υλοποίησης:

Security provider και εδώ και σε όλες τις υλοποιήσεις είναι ο BouncyCastleProvider

Η Alice χρησιμοποιεί ένα KeyPair το οποίο δημιουργείται κατα την εκτέλεση με αλγόριθμο RSA μήκους 2048 bits

Ο Bob δεν χρειάζεται να έχει κάποιο keypair για την υλοποίηση αυτού του πρωτοκόλλου. Δημιουργεί το Common Secret Key με αλγόριθμο AES και μήκος 256.

Η Alice για το encryption χρησιμοποιεί αλγόριθμο AES, cipher mode CBC, padding PKCS5 και το initiation vector δημιουργείται κατα την κρυπτογράφηση και προστίθεται για τη μεταφορά του σε έναν πίνακα byte μαζί με το απλό κρυπτογράφημα. Κατα την αποκρυπτογράφηση γίνεται η αντίστροφη διαδικασία.

2. Diffie-Hellman Key Agreement Protocol

Περιγραφή Πρωτοκόλλου:

Το πρωτόκολλο Diffie-Hellman περιλαμβάνει τα ακόλουθα βήματα:
(οι ακόλουθες ενέργειες εκτελούνται απ'τις 2 οντότητες - Alice & Bob)

Alice: διαλέγει 2 primes g και p, ως κοινές παραμέτρους (μια εκ τις οποίες είναι ο modulus που έχει μήκος 2048 για ασφάλεια).
Alice: διαλέγει έναν τρίτο prime x (το ιδιωτικό κλειδί της).
Alice: υπολογίζει το g^x (το δημόσιο κλειδί της).
Alice: στέλνει στον Bob τις κοινές παραμέτρους, g και p που διάλεξε, μαζί με το δημόσιο κλειδί της.
Bob: διαλέγει και αυτός με τη σειρά του έναν prime y (το ιδιωτικό κλειδί του).
Bob: υπολογίζει το g^y (το δημόσιο κλειδί του) .
Bob: υπολογίζει το κοινό συμμετρικό κλειδί $K = (g^y)^x \bmod p$.
Bob: στέλνει το δημόσιο κλειδί του.
Alice: υπολογίζει το κοινό συμμετρικό κλειδί $K = (g^x)^y \bmod p$.

Alice: κρυπτογραφεί ένα μήνυμα με το συμμετρικό κλειδί που διαθέτει πλέον.
Bob: αποκρυπτογραφεί με τη σειρά του το μήνυμα με το κοινό (πλέον) συμμετρικό κλειδί.

Τεχνικές Λεπτομέρειες Υλοποίησης:

Για τη δημιουργία των παραμέτρων του Diffie-Hellman, υλοποιήσαμε 3 τεχνικές.

Χρήση της κλάσσης AlgoritmParametersGenerator. (most secure but slow)
Χρήση της μεθόδου propablePrime της κλάσσης BigInteger.
Χρήση των προκαθορισμένων primes βάση του προτύπου RFC3526.

Η υλοποίηση έγινε με τη χρήση της KeyAgreement με το υλοποιημένο instance του πρωτυποποιημένου πρωτοκόλου Diffie-Hellman.

3. Station-To-Station Key Agreement Protocol

Περιγραφή Πρωτοκόλλου:

Το πρωτόκολλο Station-To-Station Diffie-Hellman περιλαμβάνει τα ακόλουθα βήματα:
(οι ακόλουθες ενέργειες εκτελούνται απ'τις 2 οντότητες - Alice & Bob)

Και η Alice και ο Bob, εμπιστεύονται κοινά μια τρίτη οντότητα (εμάς, ως Certificate Author).

Alice: διαλέγει 2 primes g και p, ως κοινές παραμέτρους
Alice: διαλέγει έναν τρίτο prime x (το ιδιωτικό κλειδί της).
Alice: υπολογίζει το g^x (το δημόσιο κλειδί της).
Alice: στέλνει στον Bob τις κοινές παραμέτρους, g και p που διάλεξε, μαζί με το δημόσιο κλειδί της.
Bob: διαλέγει και αυτός με τη σειρά του έναν prime y (το ιδιωτικό κλειδί του).
Bob: υπολογίζει το g^y (το δημόσιο κλειδί του).
Bob: υπολογίζει το κοινό συμμετρικό κλειδί $K = (g^x)^y \bmod p$.
Bob: στέλνει το δημόσιο κλειδί του και μια κρυπτογραφημένη (με το K) ψηφιακή υπογραφή που περιέχει την υπογραφή του πάνω στα 2 δημόσια κλειδιά τους και το πιστοποιητικό του που περιλαμβάνει τις παραμέτρους Diffie-Hellman και έχει υπογραφεί απ'την CA).
Alice: παραλαμβάνει το κρυπτογραφημένο πιστοποιητικό του Bob καθώς και το δημόσιο κλειδί του και ελέγχει την αυθεντικότητα των στοιχείων του.
Alice: υπολογίζει το κοινό συμμετρικό κλειδί $K = (g^y)^x \bmod p$.
Alice: αποκρυπτογραφεί με το K την ψηφιακή υπογραφή του Bob και την επιβεβαιώνει με το δημόσιο κλειδί της και του Bob καθώς και τις παραμέτρους DH που βρίσκονται στο πιστοποιητικό σε σχέση με τις δικές της. Τέλος επιβεβαιώνει οτι το πιστοποιητικό είναι υπογεγραμμένο απ'τη CA.
Alice: με τον ίδιο τρόπο η Alice φτιάχνει την ψηφιακή της υπογραφή και τη στέλνει στον Bob
Bob: παραλαμβάνει και πιστοποιεί με αντίστοιχο τρόπο την αυθεντικότητα των στοιχείων της.

Alice: κρυπτογραφεί ένα μήνυμα με το συμμετρικό κλειδί που διαθέτει πλέον.
Bob: αποκρυπτογραφεί με τη σειρά του το μήνυμα με το κοινό (πλέον) συμμετρικό κλειδί.

Τεχνικές Λεπτομέρειες Υλοποίησης:

Με τον ίδιο τρόπο δημιουργούμε τις παραμέτρους του Diffie-Hellman (όπως στην προηγούμενη υλοποίηση)

Τα ζεύγη κλειδιών που χρησιμοποιούνται για τη δημιουργία των πιστοποιητικών είναι RSA και έχουν μήκος 2048. Τα πιστοποιητικά ακολουθούν το πρότυπο X509 και έχουμε προσθέσει με

custom επεκτάσεις σε αυτά τις παραμέτρους (Diffie-Hellman) με δικά μας ASN1 OIDs

Η υλοποίηση έγινε με τη χρήση της KeyAgreement με το υλοποιημένο instance του πρωτυποποιημένου πρωτοκόλου Diffie-Hellman.

4. Ευπάθειες & Ενδυνάμωση

Ευπάθειες:

Το EKAP είναι vulnerable σε επιθέσεις Man in the middle και Replay attacks καθώς δεν παρέχει καμία αξιοπιστία για την ακεραιότητα των δεδομένων που μεταφόρονται, ούτε και για την πιστοποίηση των οντοτήτων που επικοινωνούν.

Το Diffie-Hellman είναι vulnerable σε επιθέσεις man in the middle και replay attacks καθώς ούτε αυτό πιστοποιεί τις οντότητες που επικοινωνούν και έτσι μια αντίπαλη οντότητα η οποία μπορεί να επεξεργαστεί τα αρχικά μηνύματα που αποστέλλονται μεταξύ των 2 οντοτήτων, εύκολα μπορεί να ξεγελάσει τις οντότητες προκειμένου να δημιουργήσει ένα συμμετρικό κλειδί με την κάθε μια και υποδυόμενη κάθε φορά την άλλη οντότητα να χειραγωγεί την επικοινωνία μεταξύ των 2.

Το Station-To-Station ουσιαστικά είναι μια ενδυναμωμένη έκδοση του πρωτοκόλου Diffie-Hellman καθώς γίνεται χρήση πιστοποιητικών, επομένως δεν είναι vulnerable σε επιθέσεις man in the middle, ωστόσο, είναι σε επιθέσεις Logjam, κατα τις οποίες ο επιτηθέμενος, σκοπό του έχει να υποβαθμίσει την ασφάλεια της επικοινωνίας.

Ενδυνάμωση:

Για να ενδυναμώσουμε το πρωτόκολο Station-To-Station εναντίον Logjam επιθέσεων, χρησιμοποιήσαμε ελλειπτικές καμπύλες στο πρωτόκολο Diffie-Hellman για την δημιουργία των κλειδιών του. Κάναμε χρήση των default παραμέτρων (Elliptic Curve). Μια αναλυτικότερη προεσκόπιση του υπάρχει με τη χρήση του τέταρτου κουμπιού στο GUI (SUPER kap).

EKAP

1)Alice generates her RSA KeyPair...

Alice's KeyPair loaded:

Algorithm: RSA
Format: PKCS#8
Public Key Size: 2352 bits
Private Key Size: 9736 bits

Alice's Public Key:
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtGJKeVMeu/si5+WuN9sRJpK9uTxydnLvbG+vkMISNo1ZSGNXqF5k09gGDgCJ

Alice's Private Key:
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQC0Ykp5Ux67+yLn5a432xEmkr25PHJ2cu9sb6+QwhI2jVlIY1eoXmTT

2) Alice encodes her RSA public key, and sends it to Bob...

Alice -> Bob: Alice's PublicKey Bytes

3) Bob generates the Common SecretKey they are going to use...

Common SecretKey generated:

Algorithm: AES
Format: RAW
Size: 256 bits

Common SecretKey:
/LVph8VCEj7WHAECa15KqQcmQWZpOFIGaL2R2Nv5AhQ=

4) Bob encodes the Common SecretKey he generated, encrypts it with Alice's public key, and sends it to Alice...

Bob -> Alice:

Encoded Common SecretKey:
JSzXGYyxKdVhCAnotPQS7iOxWuTjIdtlUYP0II46RRlU7Rqb2dG/smUjhlIedA5QRLOctwle36oYfClR+6YgBnQfL3RdbaBhjT3DNFWl

5) Alice receives the Encrypted Common SecretKey, decrypts and reconstructs it...

Common SecretKey decrypted and reconstucted:

Algorithm: AES
Format: RAW
Size: 256 bits

6) Alice encrypts with the Common Secret Key, using AES in CBC mode with PKCS#5 Padding, a message then encodes it and sends it to Bob...

Alice -> Bob: [encrypted message encoded]

7) Bob decrypts with the Common Secret Key, using the same algorithm, mode and padding, the encrypted message Alice sent him...

Decrypted message: This is ma big secret

DH 1/2

```
1) Alice generates the Common DiffieHellman Parameters...

        DiffieHellman parameters generated:

P (modulus): 323170060713110073003389139264238282488179412411402391128420097514007417066343542226196894173
G (generator): 2
L (exponent size): 2047 bits


2) Alice generates her DiffieHellman KeyPair using those parameters...

        Alice's KeyPair generated:

Algorithm: DH
Format: PKCS#8
Public Key Size: 4456 bits
Private Key Size: 4464 bits

Alice's Public Key:
MIICKTCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+VGbI

Alice's Private Key:
MIICKgIBADCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+

3) Alice initializes her DiffieHellman KeyAgreement instance with her private key...

        Alice's DiffieHellman KeyAgreement initialized


4) Alice encodes her DHPublicKey (including the common DH parameters), and sends it to Bob...

        Alice -> Bob: Alice's PublicKey Bytes


5) Bob receives Alice's encoded DHPublicKey and reconstructs it...

Alice's PublicKey reconstructed:

Alice's Public Key:
MIICKTCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+VGbI


6) Bob gets the Common DiffieHellman Parameters Alice generated from her DHPublicKey..

        DiffieHellman parameters from key:

P: 323170060713110073003389139264238282488179412411402391128420097514007417066343542226196894173635693471171

G: 2

L: 2047


7) Bob generates his DiffieHellman KeyPair using those parameters...

        Bob's KeyPair generated:

Algorithm: DH
Format: PKCS#8
```

```
Bob's Public Key:
MIICKDCCARsGCSqGSIb3DQEDATCCAQwCggEBAP/////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+

Bob's Private Key:
MIICKgIBADCCARsGCSqGSIb3DQEDATCCAQwCggEBAP/////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jj(

8) Bob initializes his DiffieHellman KeyAgreement with his private key...

        Bob's DiffieHellman KeyAgreement initialized

9) Bob generates the Common Secret Key using Alice's public key and his private key...

        Bob's Common SecretKey generated:

Algorithm: AES
Format: RAW
Size: 256 bits

Bob's Common SecretKey:
GYxUBGgp4yFGnjYZIOX2HcwkCIlK4zyOi9pdNqtipDQ=

10) Bob encodes his DH public key, and sends it to Alice...

        Bob -> Alice: Bob's PublicKey Bytes

11) Alice receives Bob's encoded DH public key and reconstructs it...

        Bob's PublicKey reconstructed:

Bob's Public Key:
MIICKDCCARsGCSqGSIb3DQEDATCCAQwCggEBAP/////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+

12) Alice generates the Common Secret Key usgin Bob's public key and her private key...

        Alice's Common SecretKey generated:

Algorithm: AES
Format: RAW
Size: 256 bits

Alice's Common SecretKey:
GYxUBGgp4yFGnjYZIOX2HcwkCIlK4zyOi9pdNqtipDQ=

13) Alice encrypts with the Common Secret Key, using AES in CBC mode with PKCS#5 Padding,
    a message then encodes it and sends it to Bob...

Alice -> Bob: [encrypted message encoded]

14) Bob decrypts with the Common Secret Key, using the same algorithm, mode and padding,
    the encrypted message Alice sent him...

Decrypted message: This is ma big secret
```

```
1) Alice generates the Common DiffieHellman Parameters...

        DiffieHellman parameters generated:

P (modulus): 323170060713110073003389139264238282488179412411402391128420097514007417066343542226196894l
G (generator): 2
L (exponent size): 2047 bits


2) Alice generates her DiffieHellman KeyPair using those parameters...

        Alice's KeyPair generated:

Algorithm: DH
Format: PKCS#8
Public Key Size: 4456 bits
Private Key Size: 4464 bits

Alice's Public Key:
MIICKTCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+V

Alice's Private Key:
MIICKgIBADCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE

3) Alice initializes her DiffieHellman KeyAgreement instance with her private key...

        Alice's DiffieHellman KeyAgreement initialized

4) Alice encodes her DHPublicKey (including the common DH parameters), and sends it to Bob...

        Alice -> Bob: Alice's PublicKey Bytes

5) Bob receives Alice's encoded DHPublicKey and reconstructs it...

Alice's PublicKey reconstructed:

Alice's Public Key:
MIICKTCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+V

6) Bob gets the Common DiffieHellman Parameters Alice generated from her DHPublicKey..

        DiffieHellman parameters from key:

P: 323170060713110073003389139264238282488179412411402391128420097514007417066343542226196894173635693471

G: 2

L: 2047

7) Bob generates his DiffieHellman KeyPair using those parameters...

        Bob's KeyPair generated:

Algorithm: DH
Format: PKCS#8
```

STS 1/5

Bob's Public Key:
MIICKDCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+VGb

Bob's Private Key:
MIICKgIBADCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e

8) Bob initializes his DiffieHellman KeyAgreement with his private key...

Bob's DiffieHellman KeyAgreement initialized

9) Bob generates the Common Secret Key using Alice's public key and his private key...

Bob's Common SecretKey generated:

Algorithm: AES
Format: RAW
Size: 256 bits

Bob's Common SecretKey:
BmbdK5q/VaNkNO52GUjHZLGZtx73GyNTng4HjZ4csUI=

10) Bob generates a pair of RSA keys (for his signature)...

Bob's RSA KeyPair generated:

Algorithm: RSA
Format: PKCS#8
Public Key Size: 2352 bits
Private Key Size: 9736 bits

Bob's Public Key:
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAvqQJkV52bW/3AYhPmZxFn5wyLKyK18WN4myaMAQgk5ObUFxi8N37j2bMPHZRrA

Bob's Private Key:
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQC+pAmRXnZtb/cBiE+ZnEWfnDIsrIrXxY3ibJowBCCTk5tQXGLw3fuPZs

11) Bob generates a X509 Certificate Signed by the CA including his name,
his RSA public key and the common DH parameters...

Bob's Certificate generated:

Type: X.509
Issuer: CN=MultiKAP CA,O=Χρήστος Αυλακιώτης 321|2012015 - Νίκος Τρίτσης 321|2011162
Issued to: CN=Bob
Expiration: Wed May 09 00:31:26 EEST 2018

Bob's Certificate:
MIIGGjCCBQKgAwIBAgIGAV+dwm8kMA0GCSqGSIb3DQEBBQUAMHgxFDASBgNVBAMMC011bHRpS0FQIENBMWAwXgYDVQQKDFfOp8+Bzq7Pg8+

12) Bob signs his and Alice's DH public keys with his DH private key (including his certificate)...

Bob's Signature generated:

Signature Algorithm: SHA1withRSA

Bob's Signature:
MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIAwggYaMIIFAqADAgECAgYBX53CbyQwDQYJKoZIh

13) Bob encrypts his signature with the Common Secret Key, using AES in ECB mode with PKCS#5 Padding...

Bob's Signature encrypted:

Encrypted Signature:
+XfJrOGJDsTEq888R3ZOB562klX+nv3cE4xU2YP4XLLYO9/uRcpc+FfNXZkoF70tVTW6VlQ/G02FOL05cmqq6+Bsp5mRo88Hl7kfR9o7h

14) Bob sends his encoded DH public key and his encrypted signature of the DH public keys (including his

Bob -> Alice: Bob's PublicKey Bytes, Bob's Signature

15) Alice receives Bob's signature and encoded DH public key and reconstructs it...

Bob's PublicKey reconstructed:

Bob's Public Key:
MIICKDCCARsGCSqGSIb3DQEDATCCAQwCggEBAP//////////yQ/aoiFowjTExmKLgNwc0SkCTgiKZ8x0Agu+pjsTmyJRSgh5jjQE3e+VGb

16) Alice generates the Common Secret Key usgin Bob's public key and her private key...

Alice's Common SecretKey generated:

Algorithm: AES
Format: RAW
Size: 256 bits

Alice's Common SecretKey:
BmbdK5q/VaNkNO52GUjHZLGZtx73GyNTng4HjZ4csUI=

17) Alice decrypts the signature using the same algorithm, mode and padding and extracts Bob's Certificat

Bob's signature decrypted:

Bob's Decrypted signature: MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIAwggYaMIIFAq

Bob's certificate extracted:

Type: X.509
Issuer: CN=MultiKAP CA,O=Χρήστος Αυλακιώτης 321|2012015 - Νίκος Τρίτσης 321|2011162
Issued to: CN=Bob
Expiration: Wed May 09 00:31:26 EEST 2018

Bob's Certificate:
MIIGGjCCBQKgAwIBAgIGAV+dwm8kMA0GCSqGSIb3DQEBBQUAMHgxFDASBgNVBAMMC011bHRpS0FQIENBMWAwXgYDVQQKDFfOp8+Bzq7Pg

18) Alice verifies Bob's signature over her and his DH public keys and that the DH parameters
    included in his certificate are those she sent him...
        Verification Results:

Bob's signature verified: true
Bob's DH parameters verified: true

19) Alice generates a pair of RSA keys (for her signature)...

        Alice's RSA KeyPair generated:

Algorithm: RSA
Format: PKCS#8
Public Key Size: 2352 bits
Private Key Size: 9736 bits

Bob's Public Key:
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAndDtpRWKXOg5nduommLNU6dOdNWbolQJ+4WOLWUV3fHi22Jpd4r5CN3xAVMSw0S

Bob's Private Key:
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQCd0O21FYpc6Dmd26iaYslTp0501ZuiVAn7hY4tZRXd8eLbYml3ivkI3fE

20) Alice generates a X509 Certificate Signed by the CA including her name,
    her RSA public key and the common DH parameters...
        Bob's Certificate generated:

Type: X.509
Issuer: CN=MultiKAP CA,O=Χρήστος Αυλακιώτης 321|2012015 - Νίκος Τρίτσης 321|2011162
Issued to: CN=Alice
Expiration: Wed May 09 00:31:27 EEST 2018

Bob's Certificate:
MIIGGjCCBQKgAwIBAgIGAV+dwm8kMA0GCSqGSIb3DQEBBQUAMHgxFDASBgNVBAMMC011bHRpS0FQIENBMWAwXgYDVQQKDFfOp8+Bzq7Pg8+

21) Alice signs her and Bob's DH public keys with her DH private key (including her certificate)...

        Alice's Signature generated:

Signature Algorithm: SHA1withRSA

Alice's Signature:
MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIAwggYaMIIFAqADAgECAgYBX53CbyQwDQYJKoZIhvcl

22) Alice encrypts her signature with the Common Secret Key, using AES in ECB mode with PKCS#5 Padding...

        Alice's Signature encrypted:

Encrypted Signature:
+XfJrOGJDsTEq888R3ZOB562klX+nv3cE4xU2YP4XLLYO9/uRcpc+FfNXZkoF70tQO2M0SLFHj0axnpMUyL72+rYj8HiTzt4LfTXb+E0tLal

23) Alice sends her encrypted signature of the DH public keys (including her certificate) to Bob...

        Alice -> Bob: Alice's Signature

```
24) Bob receives and decrypts Alice's signature using the same algorithm,
    mode and padding and extracts her Certificate...
        Alice's signature decrypted:

Alice's Decrypted signature: MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIAwggYcMIIFBD

        Alice's certificate extracted:

Type: X.509
Issuer: CN=MultiKAP CA,O=Χρήστος Αυλακιώτης 321|2012015 - Νίκος Τρίτσης 321|2011162
Issued to: CN=Alice
Expiration: Wed May 09 00:31:27 EEST 2018

Alice's Certificate:
MIIGHDCCBQSgAwIBAgIGAV+dwnKDMA0GCSqGSIb3DQEBBQUAMHgxFDASBgNVBAMMC011bHRpS0FQIENBMWAwXgYDVQQKDFfOp8+Bzq7Pg8+

25) Bob verifies Alice's signature over his and her DH public keys and that the
    DH parameters included in her certificate are those she sent him...
        Verification Results:

Bob's signature verified: true
Bob's DH parameters verified: true

26) Alice encrypts with the Common Secret Key, using AES in CBC mode with PKCS#5 Padding,
    a message then encodes it and sends it to Bob...
Alice -> Bob: [encrypted message encoded]

27) Bob decrypts with the Common Secret Key, using the same algorithm, mode and padding,
    the encrypted message Alice sent him...
Decrypted message: This is ma big secret
```

6. Πηγές

1. Springer Encyclopedia of Cryptography and Security, Krzysztof Kryszczuk, January 2011, ResearchGate
2. New Directions in Cryptography WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEM BER, IEEE TRANSACTIONS ON INFORMATION THEORY, NOVEMBER 1976
3. Diffie–Hellman Wikipedia page - https://en.wikipedia.org/wiki/Diffie%E2%80%93Hell man_key_exchange
4. Diffie-Hellman: Key Exchange and Public Key Cryptosystems, Sivanagaswathi Kallam, Math and Computer Science Department Indiana State University TerreHaute,IN,USA 9/2015
5. Diffie-Hellman Key Exchange – A Non-Mathematician's Explanation, ISSA Journal, 10/2006
6. Authentication and Authenticated Key Exchanges, Whitfield Diffie, Sun Microsystems, 2550 Garcia Ave., Mountain View, CA 94043 USA 1992 March
7. Station-to-Station Wikipedia page - https://en.wikipedia.org/wiki/Station-to-Station_ protocol
8. Public key cryptography - Diffie-Hellman Key Exchange (full version), Art of the Prob lem, Youtube Video, Ιουλ 2012
9. Logjam Wikipedia page - https://en.wikipedia.org/wiki/Logjam_(computer_security)
10. Extension of STS protocol on Elliptic Curve Diffie-Hellman to reduce Man-In-The-Mid dle attack, P.Varalakshmi, A.R.Shajina, B. Vinothini, International Journal of Advances in Electronics and Computer Science, ISSN: 2393-2835, Oct.-2016
11. Replay attack Wikipedia page - https://en.wikipedia.org/wiki/Replay_attack
12. Unknown Key-Share Attacks on the Station-to-Station (STS) Protocol, Simon Blake-Wil son and Alfred Menezes
13. A (relatively easy to understand) primer on elliptic curve cryptography,Nick Sulivan, 10/2013, arstechnica.com
14. Elliptic Curve Diffie Hellman, Robert Pierce, Youtube Video, Δεκ 2014
15. Elliptic curve Wikipedia page - https://en.wikipedia.org/wiki/Elliptic_curve
16. Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice, WeakDH.org
17. SPDH – A Secure Plain Diffie–Hellman Algorithm, Tange, Henrik, Journal of Cyber Se curity and Mobility, 2012