ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
Τμήμα Πληροφορικής

Ομαδική εργασία - Ασφάλεια Δικτύων και Επικοινωνιών

Δημιουργία και εγκατάσταση πιστοποιητικών σε web server με openssl

ΕΚΦΩΝΗΣΗ ΕΡΓΑΣΙΑΣ

(Α) Υλοποίηση Αρχής Πιστοποίησης και διαχείριση πιστοποιητικών. Σε αυτή την εργασία θα πρέπει να υλοποιήσετε μία ιεραρχία Αρχών Πιστοποίησης (ΑΠ - CA) -- (Root CA και Intermediate CA) με τη χρήση του openssl, και με τη βοήθεια αυτής να δημιουργήσετε το ψηφιακό πιστοποιητικό ενός web server. Μπορείτε να χρησιμοποιήσετε το openssl απευθείας σε περιβάλλον linux (π.χ. Kali linux σε virtual box/VMWare).

1. Δημιουργία Ιεραρχικής Αρχής Πιστοποίησης και έκδοση πιστοποιητικών για τις Αρχές Πιστοποίησης: Δημιουργήστε μία κύρια (Root CA) και μία ενδιάμεση (Intermediate CA) ΑΠ. Το πιστοποιητικό της Root CA θα είναι αυτο-υπογεγραμμένο (self-signed certificate).

Για την Intermediate CA, θα δημιουργήσετε το ζεύγος κλειδιών, καθώς και ένα αίτημα πιστοποίησης (certificate signing request - csr) προς την Root CA, ώστε να υπογράψει το πιστοποιητικό της Intermediate CA. Τέλος δημιουργήστε το certificate chain που περιλαμβάνει τα πιστοποιητικά και των δύο Αρχών Πιστοποίησης.

Στο βοηθητικό υλικό [3] δίδονται για βοήθεια ενδεικτικές εντολές του openssl (help.txt), καθώς και τα openssl configuration files (root_openssl.cnf και intermediate_openssl.cnf αντίστοιχα) για τις δύο ΑΠ.

2. Δημιουργία και πιστοποίηση κλειδιών για τον web server: Στη συνέχεια δημιουργήστε τα κλειδιά του web server, καθώς και ένα αίτημα πιστοποίησης (certificate signing request - csr) προς την Intermediate CA, ώστε να υπογράψει το πιστοποιητικό του web server. Για τον web server θα δημιουργήσετε το ζεύγος κλειδιών της, χρησιμοποιώντας τον αλγόριθμο Elliptic Curve (EC). Δείτε για βοήθεια τις ενδεικτικές εντολές που βρίσκονται στο βοηθητικό υλικό.

3. Εισαγωγή πιστοποιητικού στον server: Υλοποιήστε τα αντίστοιχα βήματα με το εργαστηριακό μάθημα, ώστε να εισαγάγετε το πιστοποιητικό σε έναν web server (π.χ. Apache). Τώρα ο server σας θα χρησιμοποιεί το πιστοποιητικό του για να αυθεντικοποιείται στους client (περισσότερες οδηγίες θα βρείτε στο [2] στην ενότητα One way SSL authentication)

4. Διαμόρφωση του server για διπλή αυθεντικοποίηση: Δημιουργείστε ένα πιστοποιητικό για client και διαμορφώστε τον server σας ώστε να απαιτεί και οι client να αυθεντικοποιούνται με τη χρήση πιστοποιητικού, και όχι με απλό password. (περισσότερες οδηγίες θα βρείτε στο [2] στην ενότητα Two-way SSL authentication). Συνδεθείτε με τον server και εξηγήστε περιληπτικά τι συμβαίνει κατά τη σύνδεση.

5. Έλεγχος ασφάλειας. Χρησιμοποιείστε γνωστά εργαλεία ανίχνευσης μέσω του kali linux, ώστε να επαληθεύεστε την ασφαλή λειτουργία του ssl στον server σας (ενδεικτικά, sslscan, sslyze κτλ).

(Β) Συνδυασμός ARP spoofing, DNS spoofing και social engineering attack για παραβίαση ssl σύνδεσης.

1. Χρησιμοποιείστε σε εικονικό περιβάλλον (virtualbox, vmware κτλ) δύο μηχανήματα linux, ώστε να υλοποιήσετε την επίθεση arp spoofing σε συνδυασμό με site cloning (εργαλεία ettercap και setoolkit, όπως θα δείτε στο demo που βρίσκεται στον παρακάτω σύνδεσμο [4]). Εξηγήσετε σύντομα πως λειτουργεί η παραπάνω επίθεση.

2. Να προτείνετε και να εφαρμόσετε (όπου είναι δυνατό) μέτρα προστασίας από την παραπάνω επίθεση. Χρησιμοποιώντας δικτυακές και άλλες πηγές να εξηγήσετε συνοπτικά τα μέτρα προστασίας. Αναφέρετε ποιά από τα μέτρα προστασίας μπορούν να εφαρμοστούν στη μεριά του client και ποια στην μεριά του server. (Ενδεικτικά αναφέρονται: static arp, HSTS, certificate pinning κτλ.)

ΠΙΑΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1 Εισαγωγή

Η παρούσα εργασία έχει στόχο την πρακτική εξοικείωση με την υλοποίηση και διαχείριση ψηφιακών πιστοποιητικών και την εφαρμογή μέτρων ασφαλείας σε περιβάλλον Linux. Συγκεκριμένα, αναπτύχθηκε μια ιεραρχική Αρχή Πιστοποίησης (Root και Intermediate CA), δημιουργήθηκαν πιστοποιητικά για έναν web server και έναν client, και εφαρμόστηκε η αμφίδρομη αυθεντικοποίηση (mutual TLS) με χρήση SSL/TLS. Η εργασία επικεντρώνεται επίσης στην αξιολόγηση της ασφάλειας του web server μέσω δοκιμών σύνδεσης και ελέγχου κρυπτογραφικών πρωτοκόλλων και cipher suites.

Στο δεύτερο μέρος της εργασίας, μελετήθηκε η ευπάθεια του συστήματος σε επιθέσεις κοινωνικής μηχανικής και δικτυακής παρεμβολής, με έμφαση στην επίθεση ARP spoofing σε συνδυασμό με site cloning, χρησιμοποιώντας εργαλεία όπως το SEToolkit και το Ettercap. Παράλληλα, προτάθηκαν και αναλύθηκαν μέτρα προστασίας, τόσο σε επίπεδο client όσο και server, για την πρόληψη υποκλοπής διαπιστευτηρίων και την ενίσχυση της εμπιστοσύνης των ψηφιακών επικοινωνιών.

Η εργασία επιδιώκει να δείξει τη σημασία της σωστής διαχείρισης πιστοποιητικών, της εφαρμογής σύγχρονων πρωτοκόλλων ασφαλείας και της συνδυαστικής χρήσης τεχνικών προστασίας σε περιβάλλοντα όπου οι χρήστες και οι υπηρεσίες επικοινωνούν μέσω διαδικτύου.

2 Δημιουργία Ιεραρχικής Αρχής Πιστοποίησης

2.1 Σκοπός

Σκοπός του πρώτου μέρους της εργασίας είναι η υλοποίηση μίας ιεραρχικής Αρχής Πιστοποίησης (Public Key Infrastructure – PKI), η οποία αποτελείται από:

- Root CA (Κύρια Αρχή Πιστοποίησης)
- Intermediate CA (Ενδιάμεση Αρχή Πιστοποίησης)

Η Root CA αποτελεί το ανώτατο επίπεδο εμπιστοσύνης και χρησιμοποιείται αποκλειστικά για την υπογραφή της Intermediate CA.
Η Intermediate CA χρησιμοποιείται για την υπογραφή πιστοποιητικών τελικών οντοτήτων (web server και client).

Η χρήση ενδιάμεσης CA αυξάνει την ασφάλεια, καθώς σε περίπτωση συμβιβασμού της Intermediate CA, δεν απαιτείται ανάκληση της Root CA.

2.2 Δημιουργία δομής αρχείων Root CA

Για την οργάνωση των αρχείων της Root CA δημιουργήθηκε η παρακάτω δομή καταλόγων:

- `mkdir -p /home/kali/ca/root/certs`
- `mkdir -p /home/kali/ca/root/crl`
- `mkdir -p /home/kali/ca/root/newcerts`
- `mkdir -p /home/kali/ca/root/private`

Επεξήγηση εντολών:

- **mkdir** : δημιουργεί νέο κατάλογο
- **-p** : δημιουργεί και τους ενδιάμεσους καταλόγους αν δεν υπάρχουν
- **certs** : αποθήκευση πιστοποιητικών
- **crl** : αποθήκευση Certificate Revocation Lists
- **newcerts** : αποθήκευση πιστοποιητικών που εκδίδει η CA
- **private** : αποθήκευση ιδιωτικών κλειδιών

Στη συνέχεια περιορίστηκε η πρόσβαση στον φάκελο των ιδιωτικών κλειδιών:

- `chmod 700 /home/kali/ca/root/private`

Επεξήγηση εντολής:

- **chmod 700** : πρόσβαση μόνο στον ιδιοκτήτη (ανάγνωση, εγγραφή, εκτέλεση)

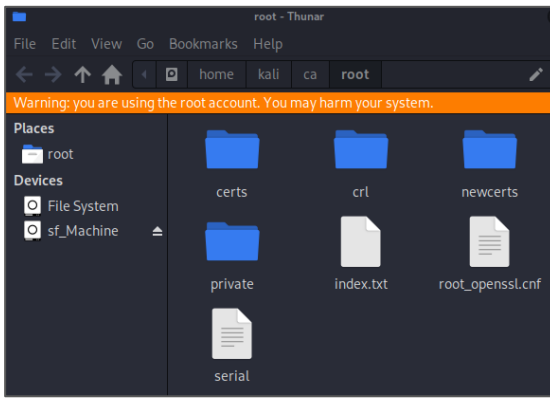
Δημιουργήθηκαν επίσης τα απαραίτητα αρχεία διαχείρισης της CA:

- `touch /home/kali/ca/root/index.txt`
- `echo 1000 > /home/kali/ca/root/serial`

Επεξήγηση αρχείων:

- **index.txt** : βάση δεδομένων εκδοθέντων πιστοποιητικών
- **serial** : αρχικός σειριακός αριθμός πιστοποιητικών

Τέλος, προστέθηκε το αρχείο root_openssl.cnf.



Δομή αρχείων ca/root

2.3 Δημιουργία ιδιωτικού κλειδιού Root CA

Για τη δημιουργία του ιδιωτικού κλειδιού της Root CA εκτελέστηκε η παρακάτω εντολή: (Εικόνα 2)

- `openssl genpkey -algorithm RSA -out /home/kali/ca/root/private/root.key.pem -aes256 -pkeyopt rsa_keygen_bits:4096`

Επεξήγηση της εντολής:

- **openssl:** Εκτελεί το εργαλείο OpenSSL, το οποίο χρησιμοποιείται για κρυπτογραφικές λειτουργίες, δημιουργία κλειδιών και πιστοποιητικών.
- **genpkey:** Υποεντολή του OpenSSL για δημιουργία ιδιωτικών κλειδιών (γενικής μορφής – πιο σύγχρονη από genrsa).
- **-algorithm RSA:** Ορίζει ότι το κλειδί που θα δημιουργηθεί θα βασίζεται στον αλγόριθμο RSA.
- **-out /home/kali/ca/root/private/root.key.pem:** Καθορίζει το αρχείο στο οποίο θα αποθηκευτεί το ιδιωτικό κλειδί.Το αρχείο αποθηκεύεται στον φάκελο private της Root CA και έχει κατάληξη .pem.
- **-aes256:** Δηλώνει ότι το ιδιωτικό κλειδί θα κρυπτογραφηθεί με τον αλγόριθμο AES-256, ώστε να προστατεύεται με συνθηματικό (password).
- **-pkeyopt rsa_keygen_bits:4096:** Ορίζει το μήκος του RSA κλειδιού σε 4096 bits, το οποίο θεωρείται ισχυρό και κατάλληλο για χρήση σε Αρχές Πιστοποίησης.

Κατά την εκτέλεση ορίστηκε κωδικός: **rootpass123**

Εικόνα 2: Δημιουργία του ιδιωτικού κλειδιού της Root CA

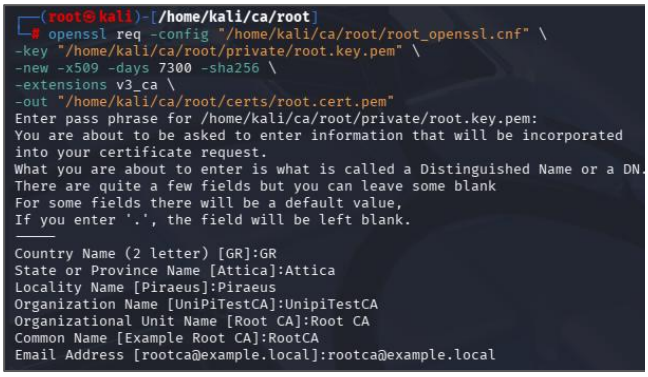
2.4 Δημιουργία self-signed πιστοποιητικού Root CA

Στη συνέχεια δημιουργήθηκε το αυτο-υπογεγραμμένο πιστοποιητικό της Root CA: (Εικόνα 3)

- ```
  openssl req -config /home/kali/ca/root/root_openssl.cnf \
  -new -x509 -days 7300 -sha256 \
  -key /home/kali/ca/root/private/root.key.pem \
  -extensions v3_ca \
  -out /home/kali/ca/root/certs/root.cert.pem
  ```

Επεξήγηση εντολής:

- **req:** Δημιουργία αιτήσεων και πιστοποιητικών
- **-config:** Αρχείο ρυθμίσεων OpenSSL
- **-new:** Δημιουργία νέου πιστοποιητικού
- **-x509:** Δημιουργία self-signed πιστοποιητικού
- **-days 7300:** Διάρκεια ισχύος (20 έτη)
- **-sha256:** Αλγόριθμος κατακερματισμού
- **-key:** Ιδιωτικό κλειδί υπογραφής
- **-extensions v3_ca:** Δηλώνει ότι πρόκειται για CA
- **-out:** Αρχείο αποθήκευσης πιστοποιητικού



Εικόνα 3: Δημιουργία self-signed πιστοποιητικού Root CA
  ```

2.5 Δημιουργία δομής αρχείων Intermediate CA

Για την οργάνωση των αρχείων της Ενδιάμεσης Αρχής Πιστοποίησης (Intermediate CA) δημιουργήθηκε η παρακάτω δομή καταλόγων:

- `mkdir -p /home/kali/ca/intermediate/certs`
- `mkdir -p /home/kali/ca/intermediate/crl`
- `mkdir -p /home/kali/ca/intermediate/newcerts`
- `mkdir -p /home/kali/ca/intermediate/private`
- `mkdir -p /home/kali/ca/intermediate/csr`

Επεξήγηση καταλόγων:

- **certs:** Αποθήκευση πιστοποιητικών
- **crl:** Αποθήκευση Certificate Revocation Lists
- **newcerts:** Αποθήκευση πιστοποιητικών που εκδίδει η Intermediate CA
- **private:** Αποθήκευση ιδιωτικών κλειδιών
- **csr:** Αποθήκευση αιτημάτων πιστοποίησης (CSR)

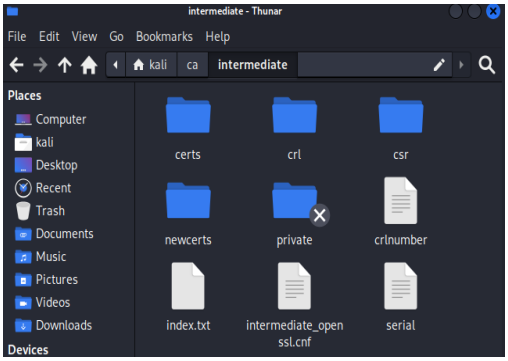
Στη συνέχεια περιορίστηκε η πρόσβαση στον φάκελο των ιδιωτικών κλειδιών:

- `chmod 700 /home/kali/ca/intermediate/private`

Δημιουργήθηκαν επίσης τα απαραίτητα αρχεία διαχείρισης της CA:

- `touch /home/kali/ca/intermediate/index.txt`
- `echo 1000 > /home/kali/ca/intermediate/serial`

Τέλος, προστέθηκε το αρχείο ρυθμίσεων intermediate_openssl.cnf.



Δομή αρχείων ca/intermidiate

2.6 Δημιουργία ιδιωτικού κλειδιού Intermediate CA

Για τη δημιουργία του ιδιωτικού κλειδιού της Intermediate CA εκτελέστηκε η παρακάτω εντολή: (Εικόνα 5)

- ```
  openssl genpkey -algorithm RSA \
  -out /home/kali/ca/intermediate/private/intermediate.key.pem \
  -aes256 -pkeyopt rsa_keygen_bits:4096
  ```

Επεξήγηση εντολής:

- **genpkey:** Δημιουργία ιδιωτικού κλειδιού
- **RSA:** Χρήση αλγορίθμου RSA
- **4096 bits:** Ισχυρό μήκος κλειδιού κατάλληλο για CA
- **aes256:** Κρυπτογράφηση ιδιωτικού κλειδιού με συνθηματικό
- **out:** Αρχείο αποθήκευσης ιδιωτικού κλειδιού

Κατά την εκτέλεση ορίστηκε κωδικός: **interpass123**

Εικόνα 5:Δημιουργία του ιδιωτικού κλειδιού της Intermidiate CA

2.7 Δημιουργία Certificate Signing Request (CSR) για την Intermediate CA

Στη συνέχεια δημιουργήθηκε αίτημα πιστοποίησης (CSR) για την Intermediate CA, το οποίο θα υπογραφεί από τη Root CA: (Εικόνα 6)

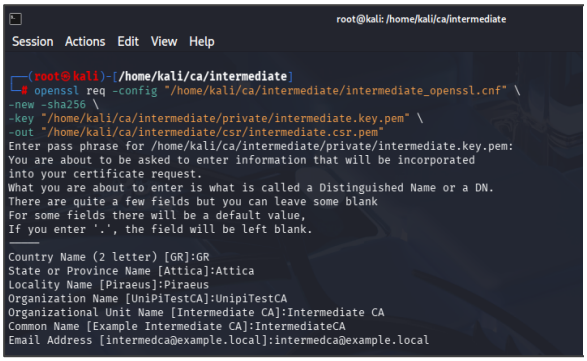
- ```
  openssl req -config
  /home/kali/ca/intermediate/intermediate_openssl.cnf \
  -new -sha256 \
  -key /home/kali/ca/intermediate/private/intermediate.key.pem \

  -out /home/kali/ca/intermediate/csr/intermediate.csr.pem
  ```

Επεξήγηση εντολής:

- **req -config:** Καθορίζει το αρχείο ρυθμίσεων που χρησιμοποιείται από το OpenSSL κατά τη δημιουργία του CSR.
- **-sha256:** Χρήση ασφαλούς αλγορίθμου κατακερματισμού
- **-key:** Ιδιωτικό κλειδί της Intermediate CA
- **-out:** Αρχείο αποθήκευσης του CSR

Το CSR περιέχει το δημόσιο κλειδί της Intermediate CA και τα στοιχεία ταυτότητάς της.



Εικόνα 6: Δημιουργία του αιτήματος πιστοποίησης (CSR) για την Intermediate CA

2.8 Υπογραφή πιστοποιητικού Intermediate CA από τη Root CA

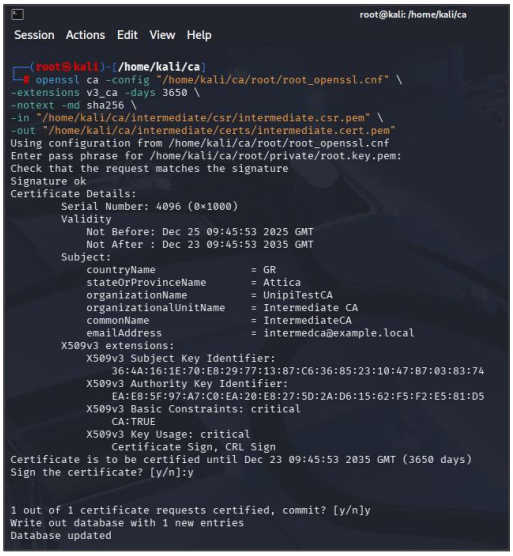
Το αίτημα πιστοποίησης της Intermediate CA υπογράφηκε από τη Root CA, δημιουργώντας το πιστοποιητικό της Intermediate CA: (Εικόνα 7)

- ```
  openssl ca -config /home/kali/ca/root/root_openssl.cnf \
  -extensions v3_ca \
  -days 3650 -notext -md sha256 \
  -in /home/kali/ca/intermediate/csr/intermediate.csr.pem \
  -out /home/kali/ca/intermediate/certs/intermediate.cert.pem
  ```

Επεξήγηση εντολής:

- **openssl ca:** Λειτουργία Αρχής Πιστοποίησης
- **-extensions v3_ca:** Δηλώνει ότι το πιστοποιητικό που δημιουργείται είναι Αρχής Πιστοποίησης (CA)
- **-days 3650:** Διάρκεια ισχύος 10 έτη
- **-md sha256:** Αλγόριθμος υπογραφής
- **-in:** CSR της Intermediate CA
  ```

- **-out:** Τελικό πιστοποιητικό Intermediate CA



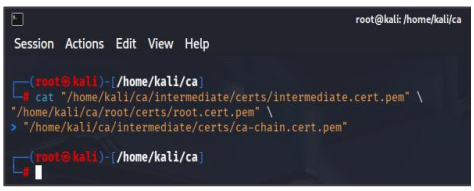
Εικόνα 7: Υπογραφή πιστοποιητικού Intermediate CA από τη Root CA

2.9 Δημιουργία Certificate Chain

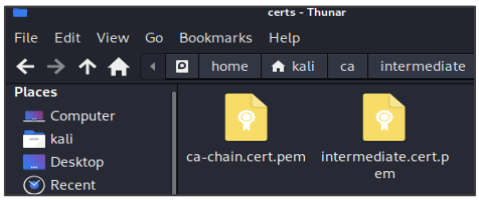
Τέλος δημιουργήθηκε το certificate chain, το οποίο περιλαμβάνει το πιστοποιητικό της Intermediate CA και το πιστοποιητικό της Root CA: (Εικόνες 8 και 9)

- `cat /home/kali/ca/intermediate/certs/intermediate.cert.pem \`
  `/home/kali/ca/root/certs/root.cert.pem \`
  `> /home/kali/ca/intermediate/certs/ca-chain.cert.pem`

Το αρχείο ca-chain.cert.pem χρησιμοποιείται από web servers και clients για την επαλήθευση της αλυσίδας εμπιστοσύνης.



Εικόνα 9: Εκτέλεση εντολής



Εικόνα 10: Εμφάνιση αρχείου chain.pem

3 Δημιουργία και πιστοποίηση Web Server

Στο επόμενο στάδιο της εργασίας δημιουργήθηκε το ζεύγος κλειδιών και το ψηφιακό πιστοποιητικό ενός web server, το οποίο υπογράφεται από την Intermediate CA. Σύμφωνα με την εκφώνηση της εργασίας, για τον web server χρησιμοποιήθηκε αλγόριθμος Elliptic Curve (EC).

3.1 Δημιουργία δομής αρχείων Web Server

Αρχικά δημιουργήθηκε ξεχωριστός φάκελος για τα αρχεία του web server:

- `mkdir -p /home/kali/server-cert`

Ο φάκελος αυτός περιέχει το ιδιωτικό κλειδί, το CSR και το τελικό πιστοποιητικό του web server.

3.2 Δημιουργία ιδιωτικού κλειδιού Web Server

Για τη δημιουργία του ιδιωτικού κλειδιού του web server εκτελέστηκε η παρακάτω εντολή: (Εικόνα 11)

- ```
  openssl genpkey -algorithm EC \
  -pkeyopt ec_paramgen_curve:prime256v1 \
  -out server.key.pem
  ```

Επεξήγηση εντολής:

- **genpkey:** Δημιουργία ιδιωτικού κλειδιού
- **-algorithm EC:** Χρήση αλγορίθμου Elliptic Curve
- **prime256v1:** Καμπύλη 256-bit, ευρέως αποδεκτή και ασφαλής
- **out:** Αρχείο αποθήκευσης ιδιωτικού κλειδιού του server

Εικόνα 11: Δημιουργία ιδιωτικού κλειδιού Web Server

3.3 Δημιουργία Certificate Signing Request (CSR) για τον Web Server

Σύμφωνα με τις σύγχρονες απαιτήσεις των browsers, το πεδίο Common Name (CN) δεν λαμβάνεται πλέον υπόψη για την επαλήθευση της ταυτότητας
  ```

ενός web server. Οι browsers ελέγχουν αποκλειστικά το πεδίο Subject Alternative Name (SAN). Για τον λόγο αυτό, δημιουργήθηκε το αρχείο server_openssl.cnf, στο οποίο ορίστηκε το πεδίο Subject Alternative Name (SAN), ώστε να ενσωματωθεί στο CSR και κατ' επέκταση στο πιστοποιητικό του server. (Εικόνα 12)

- ```
  openssl req -new \
  -key server.key.pem \
  -out server.csr.pem \
  -config server_openssl.cnf
  ```

Επεξήγηση εντολής:

- **req -new:** Δημιουργία νέου CSR
- **-key:** Ιδιωτικό κλειδί του web server
- **-out:** Αρχείο αποθήκευσης του CSR
- **-config:** Αρχείο ρυθμίσεων που περιλαμβάνει SAN

```
[ req ]
default_bits        = 2048
prompt              = no
default_md          = sha256
distinguished_name  = dn
req_extensions      = req_ext

[ dn ]
C  = GR
O  = WebServer
CN = server.lab.local

[ req_ext ]
subjectAltName = @alt_names

[ alt_names ]
DNS.1 = server.lab.local
IP.1 = 10.0.2.15
```

Εικόνα 12: Το αρχείο server_openssl.cnf

3.4 Υπογραφή πιστοποιητικού Web Server από την Intermediate CA

Το αίτημα πιστοποίησης του web server υπογράφηκε από την Intermediate CA, δημιουργώντας το τελικό πιστοποιητικό του server: (Εικόνα 13)

- ```
  openssl ca -config
  /home/kali/ca/intermediate/intermediate_openssl.cnf \
  -extensions server_cert -days 825 \
  -notext -md sha256 \
  -in /home/kali/server/server.csr.pem \
  -out /home/kali/server/server.cert.pem
  ```

Επεξήγηση εντολής:

- **openssl ca:** Λειτουργία Αρχής Πιστοποίησης
- **-extensions server_cert:** Δηλώνει ότι πρόκειται για πιστοποιητικό web server
- **-days 825:** Διάρκεια ισχύος πιστοποιητικού
- **-md sha256:** Αλγόριθμος υπογραφής
- **-in:** CSR του web server
- **out:** Τελικό πιστοποιητικό web server

Εικόνα 13: Υπογραφή πιστοποιητικού Web Server από την Intermediate CA

3.5 Τελικά αρχεία Web Server

Με την ολοκλήρωση της διαδικασίας και αφού περάσαμε το αρχείο ca-chain.cert.pem, ο web server διαθέτει τα παρακάτω αρχεία: (Εικόνα 14)

- **server.key.pem:** Ιδιωτικό κλειδί web server
- **server.cert.pem:** Πιστοποιητικό web server
- **ca-chain.cert.pem:** Αλυσίδα εμπιστοσύνης (Intermediate + Root CA)
- **server_openssl.cnf:** Αρχείο ρυθμίσεων

4 Εγκατάσταση πιστοποιητικού Web Server στον Apache (One-way SSL)

Για τη διαμόρφωση του web server ώστε να χρησιμοποιεί τα πιστοποιητικά που δημιουργήθηκαν, ακολουθήθηκαν τα παρακάτω βήματα:

4.1 Εγκατάσταση Apache

Αρχικά εγκαταστάθηκε ο Apache2:

- `sudo apt install apache2 -y`

4.2 Δημιουργία φακέλου για SSL πιστοποιητικά

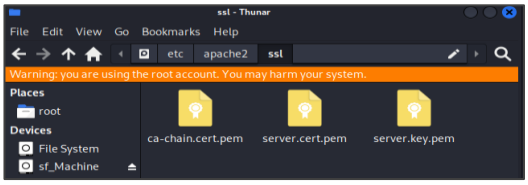
Δημιουργήθηκε ξεχωριστός φάκελος για την αποθήκευση των SSL αρχείων: (Εικόνα 14)

- `mkdir -p /etc/apache2/ssl`

Και αντιγράφηκαν εκεί τα αρχεία:

- /etc/apache2/ssl/server.cert.pem
- /etc/apache2/ssl/server.key.pem
- /etc/apache2/ssl/ca-chain.cert.pem

Ο Apache διαβάζει τα πιστοποιητικά από αυτόν τον φάκελο κατά την εκκίνηση.



Εικόνα 14: Δημιουργία φακέλου για SSL πιστοποιητικά

4.3 Ενεργοποίηση SSL στον Apache

Πρώτα ενεργοποιήθηκε το SSL module:

- `a2enmod ssl`

Μετά ενεργοποιήθηκε το SSL site:

- `a2ensite default-ssl.conf`

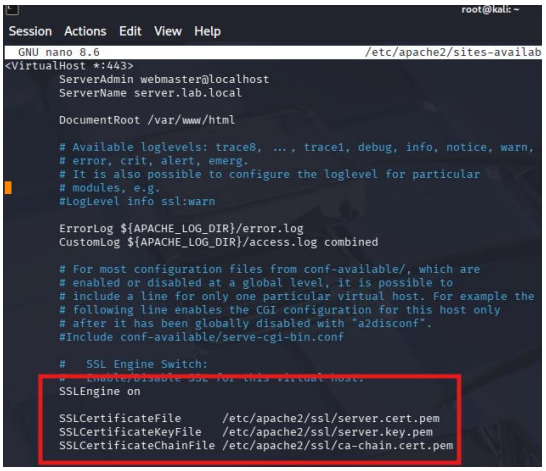
4.4 Τροποποίηση αρχείου ρυθμίσεων

Ανοίχθηκε το αρχείο:

- `nano /etc/apache2/sites-available/default-ssl.conf`

Τροποποιήθηκαν οι γραμμές SSL ώστε να χρησιμοποιούν τα νέα πιστοποιητικά: (Εικόνα 15)

- SSLEngine on
  SSLCertificateFile /etc/apache2/ssl/server.cert.pem
  SSLCertificateKeyFile /etc/apache2/ssl/server.key.pem
  SSLCertificateChainFile /etc/apache2/ssl/ca-chain.cert.pem



Εικόνα 15: Τροποποίηση αρχείου ρυθμίσεων

4.5 Επανεκκίνηση Apache και έλεγχος

Μετά τις αλλαγές:

- `systemctl restart apache2`

Έλεγχος ότι ο Apache ακούει στη θύρα 443: (Εικόνα 16)

- `ss -tlnp | grep 443`

Εικόνα 16: Δοκιμή του Apache

Ο Apache σηκώθηκε και το HTTPS είναι ενεργό.

- **apache2 ... LISTEN *:443**

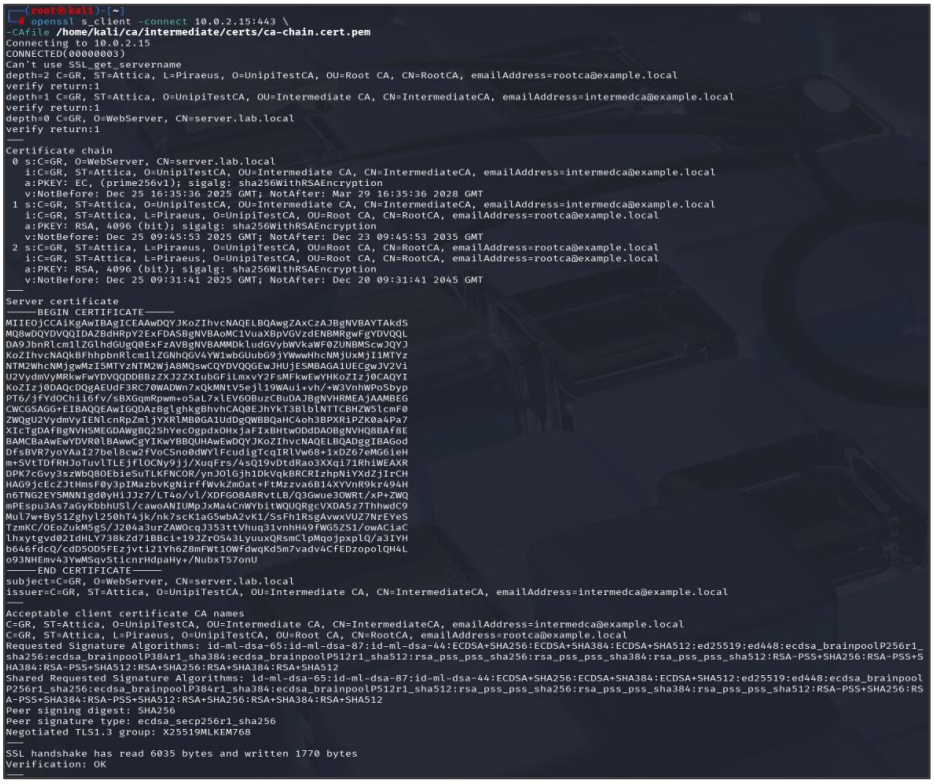
4.6 Δοκιμή σύνδεσης χωρίς client certificate

Η αρχική δοκιμή πραγματοποιήθηκε μέσω openssl s_client. Για χρήση μέσω browser απαιτήθηκε επανέκδοση του πιστοποιητικού με Subject Alternative Name (SAN) όπως είδαμε παραπάνω. Δοκιμή σύνδεσης με openssl:

- ```
  openssl s_client -connect 10.0.2.15:443 \
  -CAfile /home/kali/ca/intermediate/certs/ca-chain.cert.pem
  ```

Αποτελέσματα:

- Ο server έστειλε το πιστοποιητικό του (Εικόνα 17)
- Η αλυσίδα επαληθεύτηκε σωστά (Εικόνα 17)
- Στο τέλος εμφανίστηκε: tlsv1.3 alert certificate required (Εικόνα 18)
  ```



Εικόνα 17: Ο server έστειλε το πιστοποιητικό του και η αλυσίδα επαληθεύτηκε σωστά

Εικόνα 18: tlsv1.3 alert certificate required

Συμπέρασμα: Ο Apache δουλεύει σωστά με SSL, έχει φορτώσει το πιστοποιητικό του web server και την αλυσίδα εμπιστοσύνης της CA.
Το επόμενο βήμα αφορά τη δημιουργία client certificate και τη διαμόρφωση του server για Two-way SSL.

5 Δημιουργία και πιστοποίηση Client Certificate (Two-way SSL)

Για την ενεργοποίηση mutual TLS (two-way SSL) δημιουργήθηκε ένα client certificate και έγινε διαμόρφωση του server ώστε να απαιτεί client authentication.

Στην περίπτωσή μας, ο client προσομοιώθηκε με την εντολή openssl s_client, δηλαδή τερματικό και όχι browser, μετέπειτα προχωρήσαμε και σε browser.

5.1 Δημιουργία ιδιωτικού κλειδιού Client

Αρχικά δημιουργήθηκε φάκελος client στον φάκελο server ανοίγουμε τερματικό εκεί και έπειτα ακολουθήθηκε η ίδια διαδικασία ξεκινώντας με το ιδιωτικό κλειδί του client:

- ```
  openssl genpkey -algorithm RSA \
  -out client.key.pem \
  -pkeyopt rsa_keygen_bits:2048
  ```

Στη συνέχεια δημιουργήθηκε CSR για τον client:

- ```
  openssl req -new \
  -key client.key.pem \
  out client.csr.pem \
  -subj "/C=GR/O=Client/CN=client1"
  ```

Το CSR υπογράφηκε από την Intermediate CA:

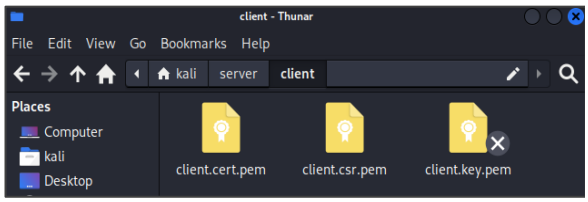
- ```
  openssl ca \
  -config /home/kali/ca/intermediate/intermediate_openssl.cnf \
  -extensions client_cert \
  -days 825 \
  -notext \
  -md sha256 \
  -in client.csr.pem \
  -out client.cert.pem
  ```
  ```

5.2 Τελική κατάσταση φακέλου client

Ο client διαθέτει τρία αρχεία: (Εικόνα 19)

- **client.key.pem:** Ιδιωτικό κλειδί
- **client.cert.pem**: Πιστοποιητικό client
- **ca-chain.cert.pem:** Αλυσίδα εμπιστοσύνης προς τον server

Χωρίς όλα τα τρία δεν γίνεται mutual TLS.



Εικόνα 19: Φάκελος client

5.3 Σύνδεση client χωρίς πιστοποιητικό

Δοκιμή σύνδεσης με client certificate και key:

```
openssl s_client \
-connect 10.0.2.15:443 \
-cert /home/kali/server/client/client.cert.pem \
-key /home/kali/server/client/client.key.pem \
-CAfile /home/kali/ca/intermediate/certs/ca-chain.cert.pem
```

Η σύνδεση ολοκληρώθηκε επιτυχώς (Εικόνα 20), η επαλήθευση του πιστοποιητικού ήταν OK (Verify return code: 0), χρησιμοποιήθηκε TLS 1.3 και διαπραγματεύτηκε σωστά το cipher. Το openssl s_client κλείνει μετά το handshake, καθώς δεν είναι browser, οπότε το closed είναι φυσιολογικό (Εικόνα 21). Με τη διαδικασία αυτή, ο server απαιτεί πιστοποιητικό client (Two-way SSL), ο client αυθεντικοποιείται μέσω του certificate χωρίς χρήση passwords, και η ταυτότητά του ελέγχεται μέσω της CA.

Εικόνα 20: Επιτυχείς σύνδεση client

Εικόνα 21 : Το openssl s_client κλείνει μετά το handshake

6 Μετατροπή Client Certificate σε PKCS#12

6.1 Λόγος μετατροπής σε PKCS#12 (.p12)

Μέχρι αυτό το στάδιο, ο client διέθετε τα παρακάτω αρχεία:

- **client.key.pem:** Ιδιωτικό κλειδί client
- **client.cert.pem:** Πιστοποιητικό client
- **ca-chain.cert.pem:** Αλυσίδα εμπιστοσύνης CA

Η μορφή αυτή (PEM αρχεία) είναι κατάλληλη για χρήση μέσω τερματικού, για εργαλεία όπως το openssl s_client και για αυτοματοποιημένα scripts. Ωστόσο, δεν είναι πρακτική για χρήση μέσω browser.

Οι σύγχρονοι browsers δεν υποστηρίζουν ξεχωριστή εισαγωγή certificate και private key, απαιτούν ένα ενιαίο αρχείο που να περιέχει ολόκληρη την ταυτότητα του client. Για τον λόγο αυτό χρησιμοποιείται το πρότυπο PKCS#12 (.p12 / .pfx), το οποίο συγκεντρώνει όλα τα απαραίτητα στοιχεία σε ένα αρχείο (το πιστοποιητικό του client, το ιδιωτικό κλειδί του και την αλυσίδα εμπιστοσύνης της CA (προαιρετικά)).

6.2 Δημιουργία αρχείου client.p12

Η μετατροπή πραγματοποιήθηκε με την παρακάτω εντολή OpenSSL:

```
openssl pkcs12 -export \
-inkey client.key.pem \
-in client.cert.pem \
-certfile /home/kali/ca/intermediate/certs/ca-chain.cert.pem \
-out client.p12
```

Επεξήγηση εντολής:

- **openssl pkcs12:** Εργαλείο OpenSSL για διαχείριση PKCS#12
- **-export:** Δημιουργία νέου αρχείου PKCS#12
- **-inkey:** Ιδιωτικό κλειδί client
- **-in:** Πιστοποιητικό client
- **-certfile:** Αλυσίδα εμπιστοσύνης CA

Κατά τη δημιουργία του αρχείου ζητήθηκε Export Password, το οποίο προστατεύει το ιδιωτικό κλειδί που περιέχεται στο αρχείο.

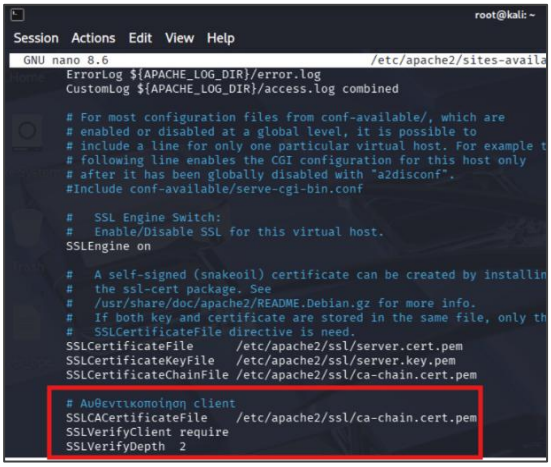
6.3 Ρύθμιση Apache για Two-way SSL

Οι σχετικές ρυθμίσεις στο Apache είναι: (Εικόνα 22)

- SSLCACertificateFile ca-chain.cert.pem
- SSLVerifyClient require
- SSLVerifyDepth 2

Με τις παραπάνω ρυθμίσεις αντίστοιχα, ο Apache:

- Επαληθεύει ότι το certificate έχει υπογραφεί από έμπιστη CA
- Απαιτεί client certificate
- Δηλώνει πόσα επίπεδα CA επιτρέπονται στην αλυσίδα εμπιστοσύνης του client



Εικόνα 22: Οι σχετικές ρυθμίσεις στο Apache

7 Ρύθμιση Client στον Browser (Two-way SSL)

Αφού δημιουργήθηκε το αρχείο client.p12, το οποίο περιέχει το πιστοποιητικό του client, το ιδιωτικό του κλειδί και την αλυσίδα εμπιστοσύνης της CA, προετοιμάστηκε για εγκατάσταση στον browser.

7.1 Προετοιμασία αρχείου client.p12

Για να μπορεί ο browser να έχει πρόσβαση στο αρχείο client.p12, στο τερματικό εκτελέστηκαν οι εντολές:

- `chown kali:kali client.p12`
- `chmod 644 client.p12`

- **chown kali:kali client.p12:** Ορίζει ιδιοκτησία του αρχείου στον χρήστη kali.
- **chmod 644 client.p12:** Επιτρέπει ανάγνωση από όλους τους χρήστες, αλλά μόνο ο ιδιοκτήτης μπορεί να γράψει.

Αυτό έγινε γιατί ο browser χρειάζεται read access στο αρχείο, ενώ δεν επιτρέπεται η εισαγωγή .p12 απευθείας χωρίς κατάλληλα δικαιώματα.

7.2 Εισαγωγή client πιστοποιητικού στον browser

Στη συνέχεια, στον browser ακολουθήθηκε η παρακάτω διαδικασία:

- Άνοιγμα ρυθμίσεων ασφαλείας
- Μετάβαση στην ενότητα Certificates → Your Certificates
- Επιλογή Import
- Επιλογή του αρχείου client.p12
- Εισαγωγή του export password

Με την ολοκλήρωση της διαδικασίας, ο browser, Εγκατέστησε το client certificate, αποθήκευσε με ασφάλεια το ιδιωτικό κλειδί και συσχέτισε το πιστοποιητικό με το αντίστοιχο private key. Επιπλέον, στις Authorities προστέθηκε η ca-chain.cert.pem ώστε ο browser να εμπιστεύεται τον issuer του client certificate.

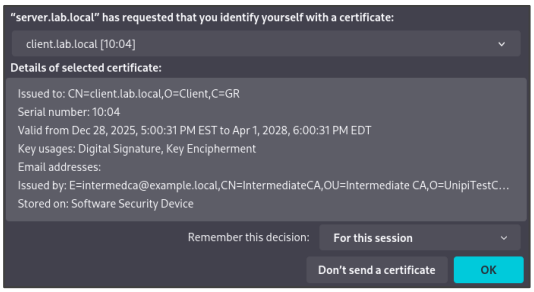
7.3 Έλεγχος λειτουργίας μέσω browser

Η δοκιμή σύνδεσης στον web server έγινε στη διεύθυνση:

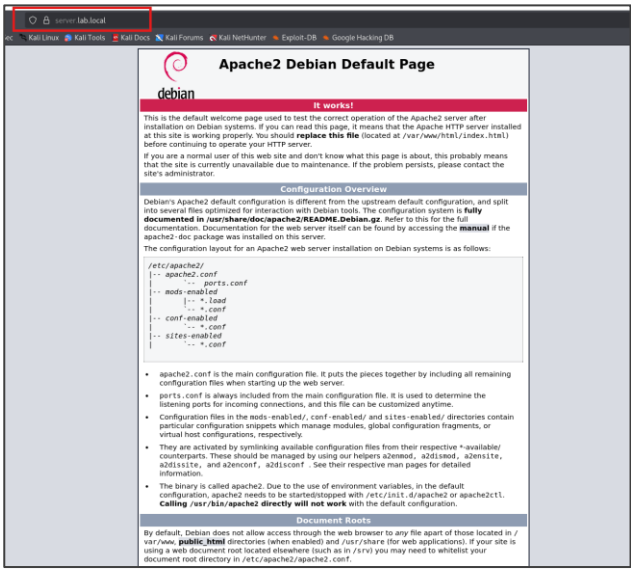
- **https://server.lab.local**

Η σωστή συμπεριφορά περιλάμβανε:

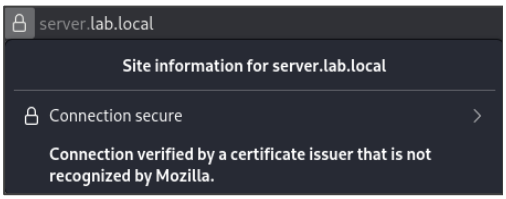
- Ο browser ζήτησε την επιλογή client certificate. (Εικόνα 23)
- Επιλέχθηκε το εγκατεστημένο client certificate.
- Ο server επαλήθευσε το πιστοποιητικό μέσω της CA.
- Η σύνδεση ολοκληρώθηκε επιτυχώς με HTTPS χωρίς προειδοποιήσεις ασφαλείας. (Εικόνες 24 και 25)



Εικόνα 23: Επιλογή client certificate



Εικόνα 24: Ολοκλήρωση σύνδεσης με HTTPS



Εικόνα 24: Ολοκλήρωση σύνδεσης με HTTPS

7.4 Έλεγχος ασφάλειας Web Server με sslscan

Για την αξιολόγηση της ασφάλειας της TLS υλοποίησης του web server πραγματοποιήθηκε έλεγχος με το εργαλείο sslscan, το οποίο επιτρέπει την ανάλυση των υποστηριζόμενων πρωτοκόλλων, αλγορίθμων κρυπτογράφησης και παραμέτρων ασφαλείας. (Εικόνα 25)

Η εντολή που χρησιμοποιήθηκε ήταν:

- `sslscan https://server.lab.local/`

Πρωτόκολλα SSL/TLS

Από τα αποτελέσματα προκύπτει ότι όλα τα παρωχημένα και ευάλωτα πρωτόκολλα (SSLv2, SSLv3, TLSv1.0 και TLSv1.1) είναι απενεργοποιημένα, ενώ ο server υποστηρίζει αποκλειστικά TLS 1.2 και TLS 1.3. Η επιλογή αυτή θεωρείται βέλτιστη πρακτική, καθώς περιορίζει σημαντικά την επιφάνεια επίθεσης και εξασφαλίζει σύγχρονη κρυπτογραφική προστασία.

Προστασία από downgrade και γνωστές επιθέσεις

Η υποστήριξη του TLS Fallback SCSV δείχνει ότι ο server προστατεύεται από downgrade attacks, ενώ η απενεργοποίηση του TLS renegotiation μειώνει τον κίνδυνο εκμετάλλευσης παλαιότερων ευπαθειών που σχετίζονται με επαναδιαπραγμάτευση συνόδου. Επιπλέον, η απενεργοποίηση της συμπίεσης TLS προστατεύει από επιθέσεις τύπου CRIME. Ο έλεγχος έδειξε επίσης ότι ο server δεν είναι ευάλωτος στην ευπάθεια Heartbleed, τόσο για TLS 1.2 όσο και για TLS 1.3.

Cipher Suites και Key Exchange

Οι υποστηριζόμενοι αλγόριθμοι κρυπτογράφησης βασίζονται αποκλειστικά σε ECDHE για ανταλλαγή κλειδιών και σε σύγχρονες AEAD σουίτες (AES-GCM και CHACHA20-POLY1305). Αυτό εξασφαλίζει Perfect Forward Secrecy, καθώς και υψηλό επίπεδο κρυπτογραφικής ισχύος (128 έως 256 bit). Οι ομάδες ανταλλαγής κλειδιών περιλαμβάνουν σύγχρονες και ασφαλείς επιλογές, όπως x25519, secp256r1, secp384r1 και ffdhe3072, οι οποίες θεωρούνται κατάλληλες για σύγχρονες TLS υλοποιήσεις.

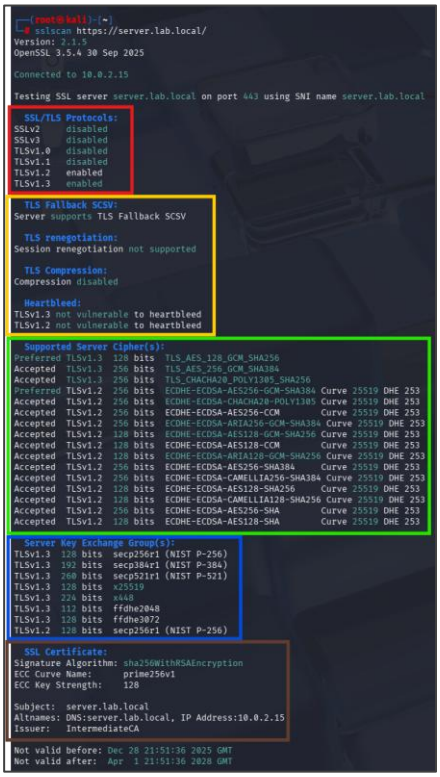
Πιστοποιητικό Web Server

Η ανάλυση του πιστοποιητικού έδειξε ότι:

- Το πιστοποιητικό έχει εκδοθεί από την Intermediate CA, σύμφωνα με την προβλεπόμενη ιεραρχία εμπιστοσύνης,
- Χρησιμοποιεί Elliptic Curve κλειδί με καμπύλη prime256v1,
- Περιλαμβάνει Subject Alternative Name (SAN) τόσο για το DNS όνομα όσο και για τη διεύθυνση IP, γεγονός που εξασφαλίζει πλήρη συμβατότητα με τους σύγχρονους browsers.

Συμπεράσματα ελέγχου

Από τον έλεγχο με το sslscan προκύπτει ότι ο web server είναι ρυθμισμένος με σύγχρονες και ασφαλείς παραμέτρους TLS, ακολουθώντας τις τρέχουσες βέλτιστες πρακτικές. Η απενεργοποίηση παλαιών πρωτοκόλλων, η χρήση ισχυρών cipher suites, η υποστήριξη Perfect Forward Secrecy και η σωστή διαμόρφωση των πιστοποιητικών επιβεβαιώνουν ότι η υλοποίηση προσφέρει υψηλό επίπεδο ασφάλειας και είναι κατάλληλη για περιβάλλοντα που απαιτούν αυξημένη προστασία, όπως η χρήση Two-way SSL (mutual TLS).



Εικόνα 25: Αποτελέσματα ελέγχου

8 Υλοποίηση επίθεσης ARP Spoofing σε συνδυασμό με Site Cloning

Στο πλαίσιο της εργασίας χρησιμοποιήθηκαν δύο εικονικές μηχανές Linux σε περιβάλλον VirtualBox:

- **Attacker VM με IP:** 192.168.1.7
- **Victim VM με IP:** 192.168.1.63

Στόχος της επίθεσης ήταν η υλοποίηση Man-in-the-Middle (MITM) επίθεσης μέσω ARP Spoofing, σε συνδυασμό με Site Cloning, με σκοπό την υποκλοπή διαπιστευτηρίων χρήστη. Η επίθεση πραγματοποιήθηκε με τα εργαλεία Ettercap και SET (Social Engineering Toolkit).

8.1 Προετοιμασία Site Cloning με SEToolkit (Attacker VM)

Στον υπολογιστή του επιτιθέμενου (192.168.1.7) εκτελέστηκε το εργαλείο setoolkit, το οποίο χρησιμοποιείται για επιθέσεις κοινωνικής μηχανικής.

Η διαδρομή που ακολουθήθηκε στο μενού ήταν:

1. Social-Engineering Attacks
2. Website Attack Vectors
3. Credential Harvester Attack Method
4. Site Cloner

Στη συνέχεια δηλώθηκε η ip του Attacker ως στόχος η ιστοσελίδα: (Εικόνα 26)

- **https://sso.unipi.gr**

Το SEToolkit δημιουργεί ένα αντίγραφο (clone) της πραγματικής ιστοσελίδας.

- Η σελίδα φιλοξενείται τοπικά στο attacker VM.
- Όλα τα credentials που εισάγονται στη φόρμα καταγράφονται σε αρχείο.

Ουσιαστικά, ο χρήστης βλέπει μία σελίδα που μοιάζει απόλυτα με την πραγματική, αλλά τα δεδομένα αποστέλλονται στον επιτιθέμενο.

Εικόνα 26: Δήλωση ip Attacker & ιστοσελίδας

8.2 Ρύθμιση DNS Spoofing στο Ettercap (Attacker VM)

Στο αρχείο ρυθμίσεων του Ettercap προστέθηκαν οι παρακάτω εγγραφές:

- **sso.unipi.gr A 192.168.1.7** # Κάθε DNS ερώτημα για το sso.unipi.gr
- ***.sso.unipi.gr A 192.168.1.7** # Καλύπτει όλους τους υποτομείς του sso.unipi.gr π.χ. login.sso.unipi.gr, auth.sso.unipi.gr
- **www.sso.unipi.gr PTR 192.168.1.7** # Το PTR record χρησιμοποιείται για reverse DNS

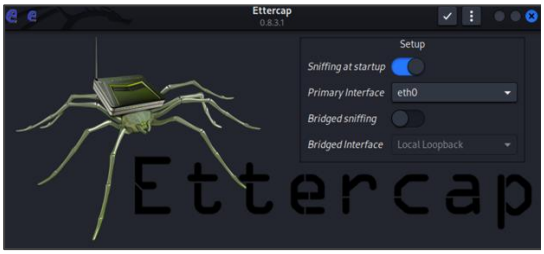
Οποιοδήποτε DNS request για το sso.unipi.gr ή υποτομείς του ανακατευθύνεται στη διεύθυνση IP του attacker αντί για τον πραγματικό server. Έτσι, το θύμα οδηγείται αυτόματα στο ψεύτικο site που φιλοξενεί το SEToolkit.

8.3 ARP Spoofing με Ettercap (Attacker VM)

Στη συνέχεια εκτελέστηκε το Ettercap σε γραφικό περιβάλλον.
Ακολουθήθηκαν τα παρακάτω βήματα:

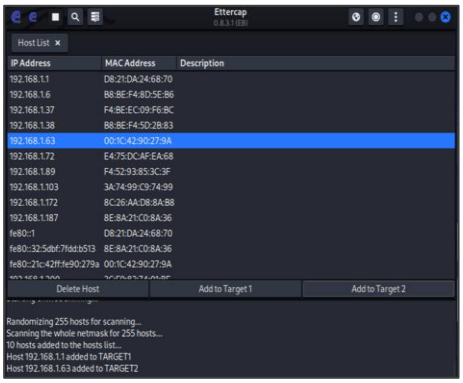
1. Ενεργοποιήθηκε το sniffing (Εικόνα 27)
2. Πραγματοποιήθηκε σάρωση για hosts (Εικόνα 28)
3. Ορίστηκαν τα targets: (Εικόνα 29)
    - **Target 1:** Gateway (192.168.1.1)
    - **Target 2:** Victim (192.168.1.63)
4. Ενεργοποιήθηκε ARP Poisoning
5. Ενεργοποίηση DNS Spoof Plugin (Εικόνα 30)

Ο attacker στέλνει ψεύτικες ARP απαντήσεις.
Έτσι, όλη η επικοινωνία περνά μέσα από τον attacker, δημιουργώντας επίθεση Man-in-the-Middle.

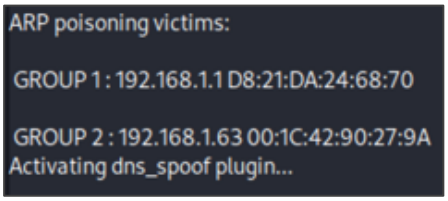


Εικόνα 27: Ενεργοποίηαη sniffing

Εικόνα 28: Σάρωση για hosts



Εικόνα 29: Ορίστηκαν τα targets



Εικόνα 30: Ενεργοποίηση DNS Spoof Plugin

8.4 Υποκλοπή διαπιστευτηρίων

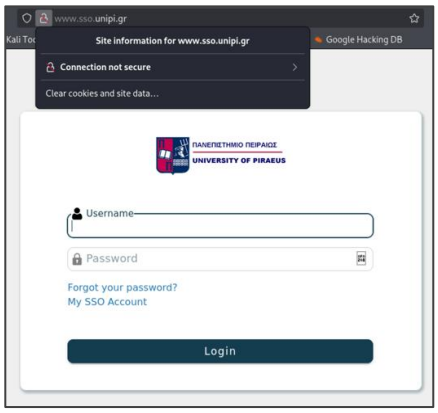
Στο Victim VM:

- Ανοίχθηκε browser
- Πληκτρολογήθηκε το: **www.sso.unipi.gr**

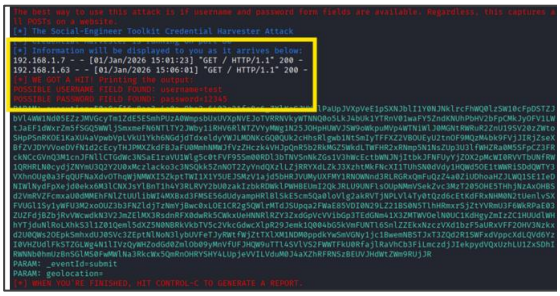
Ο χρήστης οδηγήθηκε στο cloned site του attacker (Εικόνα 31)
Με την εισαγωγή στοιχείων σύνδεσης:

- **Username**
- **Password**

Τα δεδομένα καταγράφηκαν από το SEToolkit και εμφανίστηκαν σε πραγματικό χρόνο στο Attacker VM. (Εικόνα 32)



Εικόνα 31: Cloned site του attacker



Εικόνα 3: Καταγραφή δεδομένων από το SEToolkit

9 Μέτρα προστασίας

Η επίθεση που υλοποιήθηκε βασίζεται στον συνδυασμό ARP Spoofing, DNS Spoofing και Site Cloning, επιτρέποντας την υποκλοπή διαπιστευτηρίων μέσω Man-in-the-Middle επίθεσης. Για την αντιμετώπιση τέτοιων επιθέσεων απαιτείται η εφαρμογή μέτρων προστασίας σε επίπεδο δικτύου, client και server.

9.1 Μέτρα προστασίας σε επίπεδο δικτύου

Static ARP Entries:

Η χρήση στατικών εγγραφών ARP αποτρέπει την παραποίηση των ARP απαντήσεων, καθώς η αντιστοίχιση IP–MAC δεν μπορεί να αλλαχθεί δυναμικά.

Dynamic ARP Inspection (DAI)

Μηχανισμός σε managed switches που ελέγχει τις ARP απαντήσεις και απορρίπτει τις μη έγκυρες.

9.2 Μέτρα προστασίας στη μεριά του server

HTTPS με έγκυρα πιστοποιητικά

Η χρήση TLS με πιστοποιητικά από αξιόπιστη CA είναι βασική προϋπόθεση, αλλά δεν επαρκεί από μόνη της σε περιπτώσεις DNS spoofing.

HSTS (HTTP Strict Transport Security)

Το HSTS υποχρεώνει τον browser να χρησιμοποιεί μόνο HTTPS για συγκεκριμένο domain.

Certificate Pinning

Με το certificate pinning, ο client αποδέχεται μόνο συγκεκριμένα certificates ή public keys για ένα domain.

9.3 Μέτρα προστασίας στη μεριά του client

Έλεγχος πιστοποιητικών
Ο χρήστης πρέπει να ελέγχει:

- **issuer**
- **domain name**
- **ύπαρξη έγκυρου SAN**

Εγκατάσταση μόνο έμπιστων CA
Ο client πρέπει να εμπιστεύεται μόνο γνωστές και αξιόπιστες Αρχές Πιστοποίησης.

9.4 Συμπέρασμα

Η επίθεση που υλοποιήθηκε δείχνει πώς ένας επιτιθέμενος μπορεί να υποκλέψει διαπιστευτήρια εκμεταλλευόμενος την εμπιστοσύνη του χρήστη και την πλαστογράφηση ενός νόμιμου ιστοτόπου. Παρότι η επικοινωνία φαίνεται κανονική στον χρήστη, τα δεδομένα καταλήγουν σε κακόβουλο σύστημα.

Η προστασία από τέτοιου είδους επιθέσεις δεν βασίζεται σε ένα μόνο μέτρο, αλλά στη σωστή υλοποίηση μηχανισμών ασφάλειας που διασφαλίζουν την αυθεντικότητα του εξυπηρετητή και αποτρέπουν τη χρήση πλαστών ιστοσελίδων. Η εφαρμογή αυτών των μηχανισμών μειώνει σημαντικά τον κίνδυνο υποκλοπής ευαίσθητων πληροφοριών.