Project #2
ΥΣ13 ΕΑΡΙΝΟ 2016

Αλέξανδρος Λαποκωνσταντάκης
1115201200088
(8)

Αφού έφτιαξα ένα directory ca με τα απαραίτητα αρχεία, δημιούργησα ένα CA keypair με δικά μου στοιχεία, καθώς επίσης και ένα άλλο keypair (tlsattack) με τυχαία στοιχεία. Έκανα selfsign το CA certificate και κάνοντας sign το tlsattack.csr εξέδοσα το αντίστοιχο certificate.

Create CA (selfsign)
openssl req -new -x509 -extensions v3_ca -keyout private/cakey.key -out cacert.crt -days 3650 -config ./openssl.cnf

openssl ca -gencrl -keyfile private/cakey.key -cert cacert.crt -out crl.pem -config ./openssl.cnf

Create end-entity keypair and certificate
openssl genrsa 1024 > private/tlsattack.key

openssl req -new -key private/tlsattack.key -out tlsattack.csr -config ./openssl.cnf

Sign end-entity tlsattack.crt
openssl ca -config openssl.cnf -policy policy_anything -cert cacert.crt -keyfile private/cakey.key -days 365 -out tlsattack.crt -infiles tlsattack.csr

Κάθε φορά οι clients προσπαθούν να συνδεθούν σε συγκεκριμένο port, αντί για το 1443 που είναι το port του webshop, όπου θέλουν να στείλουν τα στοιχεία τους, άρα, με την twistedeve θα κάνω bind στο port που "επιλέγουν" οι client, target στο 1443, και attack σ' ένα port που θα διαλέξω εγώ.

twistedeve -b localhost:YYYYY -t localhost:1443 -a localhost:33138 -k ca/private/tlsattack.key -c ca/tlsattack.crt

Mr Blonde
Δοκίμασα τα certificates με τα τυχαία στοιχεία και έδωσε αμέσως τον αριθμό της κάρτας του. Στη συνέχεια εξετάζοντας το private key και certificate signed από το SBOX CA, που είναι και η trusted CA των clients, είδα ότι μόνα κοινά στοιχεία που τύχαινε να έχω ήταν η πόλη και η χώρα. Αλλαζοντάς τα και αυτά, πάλι η προσπάθεια ήταν επιτυχής, πράγμα που σημαίνει ότι ο Mr Blonde δεν ελέγχει κανένα πεδίο του certificate.

Credit Card
4916281186440807&timestamp=1461869747

Mr Blue
Θα 'πρεπε να δημιουργήσω certificates, root και το certificate που θα χρησιμοποιηθεί για το attack, με στοιχεία όμοια με τα trusted των clients, έτσι, παίρνοντας απ το δοθέν signed certificate τα στοιχεία:

Issuer: C=GR, ST=ATTICA, L=Athens, O=University of Athens, OU=Department of Informatics and Telecommunications, CN=SBOX CA email=csec.di@gmail.com

Subject: C=GR, ST=ATTICA, L=Athens, O=University of Athens, OU=Department of Informatics and Telecommunications, CN=mysite.com

δημιούργησα ενα CA με στοιχεία πανομοιότυπα του issuer (SBOX CA) και αντίστοιχα ένα intermediate certificate πανομοιότυπο του subject. Έπιασε και πάλι μόνο στον Mr Blonde.

Στη συνέχεια δοκίμασα αντί για SBOX CA common name να βάλω το webshop, που είναι και ο server στον οποίο προσπαθούν να συνδεθούν. Δεν είχε αποτέλεσμα, οπότε δοκίμασα το sbox.di.uoa.gr που είναι η διεύθυνση όπου τρέχει ο server. Ο Mr Blue εμπιστεύτηκε το συγκεκριμένο common name και μπόρεσα να πάρω τα στοιχεία του. Έκανα δοκιμές για να δω τι άλλο στοιχείο ελέγχει.
Μετά από διαδοχικές δοκιμές φαίνεται ότι ελέγχει μόνο το common name του intermediate certificate, χωρίς να εξετάζει κανένα άλλο στοιχείο, ούτε το certificate chain.

Credit Card
4556521986009038&timestamp=1461888751

Mr Brown
Στη συνέχεια δοκίμασα τα δύο δοθέντα αρχεία (key και certificate) αυτούσια. Αυτή τη φορά πέτυχε και το request του Mr Brown. Φαίνεται ότι το Mr Brown, δεν τον ενδιαφέρει το common name που είναι διαφορετικό σ' αυτό το certificate (mysite.com), όμως ελέγχει αν το signature ανήκει σε trusted CA, δηλαδή ελέγχει το certificate chain, και όπως θα φανεί αργότερα, αν ο issuer του certificate είναι πραγματικά CA.

Credit Card
4539111762055348&timestamp=1461880664

Mr Orange
Η επόμενη ιδέα ήταν να δημιουργήσω ένα certificate που να είναι το τέλος ενός certificate chain που θα αρχίζει από ενα trusted (το SBOX CA), με τα υπόλοιπα στοιχεία ίδια του certificate που χρησιμοποίησα στο Mr Blue (μάλλον κάποιος απ' τους εναπομείναντες clients έλεγχε τόσο το chain, όσο και το common name, ελπίζοντας ότι δεν έλεγχε το αν ήταν CA ή όχι ο issuer).
Δοκίμασα να εκδόσω certificate απ' το tlsattack.csr που δημιούργησα (με sbox.di.uoa.gr στο CN) με τα δοθέντα αρχεία ως CA, και έπιασε. Φαίνεται ότι το mysite.com αν και δεν είναι CA κατά λάθος έχει τη δυνατότητα να εκδίδει certificates.
Έτσι τώρα το attacktls.crt ήταν signed από certificate που είναι trusted στους client, και θεωρητικά θα'πρεπε να χα ένα σχήμα certification: SBOX CA→mysite.com→ sbox.di.uoa.gr (το attacktls). Ωστόσο αυτό δεν έπιασε. Μετά από λίγο ψάξιμο φάνηκε ότι το πρόβλημα μπορεί να ήταν ότι δε γίνεται verify το certificate chain, που προσπαθώ να δημιουργήσω, και ουσιαστικά δε λειτουργεί. Έκανα το ίδιο, όμως έφτιαξα και πάλι το CA μου με τα στοιχεία του

SBOX CA, ελπίζοντας να θεωρήσει ότι ο ο πλαστός CA ήταν όντως αυτός που εξέδοσε το valid certificate, και έτσι είχα
το ίδιο chain, με ενα πλαστό SBOX CA , προφανώς με διαφορετικά κλειδιά απ' τον κανονικό που έχει όντως κάνει issue τα certificates. Αυτή τη φορά το αποτέλεσμα του twistedeve ήταν Segmentation fault. Βρήκα ότι μάλλον είχα καταλάβει λάθος το πως λειτουργούν τα chains, και εκτό από verification, πρέπει να τα δώσω ρητά και ως certificates, δεν ανιχνεύονται αυτόματα (λογικό).

Τα CAChain.pem files τα δημιουργούσα κάπως έτσι :

cat tlsattack.crt /var/project2/mysite.com.crt cacert.crt > CAChain.pem (όπου cacert.crt το certificate του δικού μου CA, και tlsattack.crt το certificate signed απ' το mysite.com, με common name το ζητούμενο sbox.di.uoa.gr)

Όταν κατάλαβα το πρόβλημα, επέστρεψα στη λύση που φαινόταν πιο δυνατή, απλώς να δημιουργήσω ένα tlsattack2.crt που να περιέχει το tlsattack.crt, signed απ' το mysite.com και το ίδιο το mysite.com.crt.

Έκανα sign εκ νέου το tlsattack.crs
openssl ca -config openssl.cnf -policy policy_anything -cert mysite.com.crt -keyfile mysite.com.key -days 365 -out tlsattack.crt -infiles tlsattack.csr

και έφτιαξα το chain: cat tlsattack.crt mysite.com.crt > CAChain.crt

Τελικά εκτέλεσα:
twistedeve -b localhost:45461 -t localhost:1443 -a localhost:33138 -k ca/private/tlsattack.key -c ca/tlsattack3.crt, το οποίο όντως δούλεψε και ο Mr Orange έδωσε την κάρτα του.

Credit Card
4716110243556374&timestamp=1462023142

Άρα φαίνεται ότι ο Mr Orange ελέγχει τόσο τα στοιχεία του certificate, όσο και το certificate chain, αλλά δεν κοιτάει αν ο issuer του certificate είναι όντως CA.

Mr Pink
Στη συνέχεια δοκίμασα να δημιουργήσω έναν openssl server που να ο οποίος υποστηρίζει μόνο DH_anon key exchange algorithm, έτσι ώστε να μην απαιτηθεί να πιστοποιήσω ότι είμαι το webshop στους clients. Εκτελώντας twisted eve με το tlsinfo.py filter φαίνεται ότι ο Mr Pink είναι ο μόνος που το υποστηρίζει, έτσι ώστε να γίνει επιτυχής σύνδεση.
(Client supports ciphersuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA', 'TLS_RSA_WITH_RC4_128_SHA', '***TLS_DH_anon_WITH_AES_256_CBC_SHA'***])

Με την εντολή openssl cipher -v 'ALL:aNULL' είδα ότι το TLS_DH_anon_WITH_AES_256_CBC_SHA αντιστοιχεί στο cipher ADH-AES256-SHA του

openssl, άρα για το στήσιμο του server έκανα:

openssl s_server -accept 4436 -key ca/private/sattack.key -nocert -cipher ADH-AES256-SHA

και έτρεξα την twistedeve:

twistedeve -b localhost: -t localhost:4436 -a localhost:33138 -k ca/private/tlsattack.key -c ca/tlsattack.crt -f /var/project2/filters/tlsinfo.py

Εκτέλεσα την επίθεση αυτή τη φορά με target όχι το webshop, αλλά το port του νέου openssl server, που υποστηρίζει μόνο DH_anon. Ο Mr Pink δέχτηκε τη σύνδεση με TLS_DH_anon_WITH_AES_256_CBC_SHA και σύμφωνα με αυτόν τον αλγόριθμο έδωσε τα στοιχεία του χωρίς να ζητήσει το certificate του server που είχα στήσει, άρα μην καταλαβαίνοντας ότι δεν πρόκειται για το webshop. Οι υπόλοιποι clients που δεν προβλέπουν το συγκεκριμμένο cipher απλώς έδοσαν:

ERROR
3083036936:error:1408A0C1:SSL routines:SSL3_GET_CLIENT_HELLO:no shared cipher:s3_srvr.c:1352:

Credit Card
4486863506852637&timestamp=1461942451

Αφού εκτέλεσα και τις υπόλοιπες επιθέσεις φάνηκε ότι ο Mr Pink εκτελεί το TLS χωρίς σφάλμα, άρα το rollback σε DH_anon ήταν η μόνη λύση.

Mr White
Ο Mr White εκτελεί σωστά το TLS, και δεν υποστηρίζει κάποιο anonymous cipher algorithm, άρα μου φαίνεται ότι θα χρειαστεί αρκετή προσπάθεια και χρόνο για να σπάσεις το TLS, που δεν είχα καθώς ουσιαστικά άρχισα την 6η μέρα παράτασης.