EAP-TTLS original

hashfunction prf;

hashfunction h;

usertype String;

const ccs, csu, success, accesslevel, clientfinished, serverfinished: String;

const ttlschallenge, ttlskeyingmaterial,mastersecret, shd, password, uname, unamenottrue: String;

macro ms= prf (pms, mastersecret, (ni, nr));

macro finishedi= prf(ms, clientfinished, h(SID, ni, csu, nr, {CerR, pk(R)}sk(CA), shd, {pms}pk(R)));

macro finisheds= prf(ms, serverfinished, h(SID, ni, csu, nr, {CerR, pk(R)}sk(CA), shd, {pms}pk(R) , finishedi));

macro clientkey= prf (ms, ttlskeyingmaterial, (ni, nr));

macro serverkey= prf (ms, ttlskeyingmaterial, (ni, nr));

macro chapchallenge= prf (ms, ttlschallenge, (ni,nr));

macro mschap= (uname, chapchallenge, password);

protocol eapttls (I, R, CA)

{

role I

{

const SID, CerR, pms, S: Data;

fresh ni: Nonce;

var nr: Nonce;

send\_1 (I, R, unamenottrue, SID, ni, csu);

recv\_4 (R, I, SID, nr, csu, {CerR, pk(R)}sk(CA), shd);

send\_5 (I, R, {pms}pk(R), ccs);

send\_6 (I, R, finishedi);

recv\_7 (R, I, ccs, finisheds);

match (prf (ms, serverfinished, h(SID, ni, csu, nr, {CerR, pk(R)}sk(CA), shd, {pms}pk(R) , finishedi)), finisheds);

send\_9 (I, R, {mschap}clientkey);

recv\_10 (R, I, {success, accesslevel} serverkey);

claim\_i1 (I, Secret, ni);

claim\_i2 (I, Secret, nr);

claim\_i3 (I, Secret, CerR);

claim\_i4 (I, Secret, ms);

claim\_i5 (I, Secret, clientkey);

claim\_i6 (I, Secret, serverkey);

claim\_i7 (I, Secret, chapchallenge);

claim\_i8 (I, Secret, password);

claim\_i9 (I, Nisynch);

claim\_i10 (I, Niagree);

}

role R

{

const SID, CerR, pms, S: Data;

fresh nr: Nonce;

var ni: Nonce;

recv\_1 (I, R, unamenottrue, SID, ni, csu);

send\_2 (R, CA, R);

recv\_3 (CA, R, {R, {CerR, pk(R)}pk(R)}sk(CA));

send\_4 (R, I, SID, nr, csu, {CerR, pk(R)}sk(CA), shd);

recv\_5(I, R, {pms}pk(R), ccs);

recv\_6 (I, R, finishedi);

match (prf (ms, clientfinished, h(SID, ni, csu, nr, {CerR, pk(R)}sk(CA), shd, {pms}pk(R))), finishedi);

send\_7(R, I, ccs, finisheds);

recv\_9 (I, R, {mschap}clientkey);

match ((uname, prf (ms, ttlschallenge, (ni,nr)), password), mschap);

send\_10 (R, I, {success, accesslevel}serverkey);

claim\_r1 (R, Secret, ni);

claim\_r2 (R, Secret, nr);

claim\_r3 (R, Secret, CerR);

claim\_r4 (R, Secret, ms);

claim\_r5 (R, Secret, clientkey);

claim\_r6 (R, Secret, serverkey);

claim\_r7 (R, Secret, chapchallenge);

claim\_r8 (R, Secret, password);

claim\_r9 (R, Nisynch);

claim\_r10 (R, Niagree);

}

role CA

{

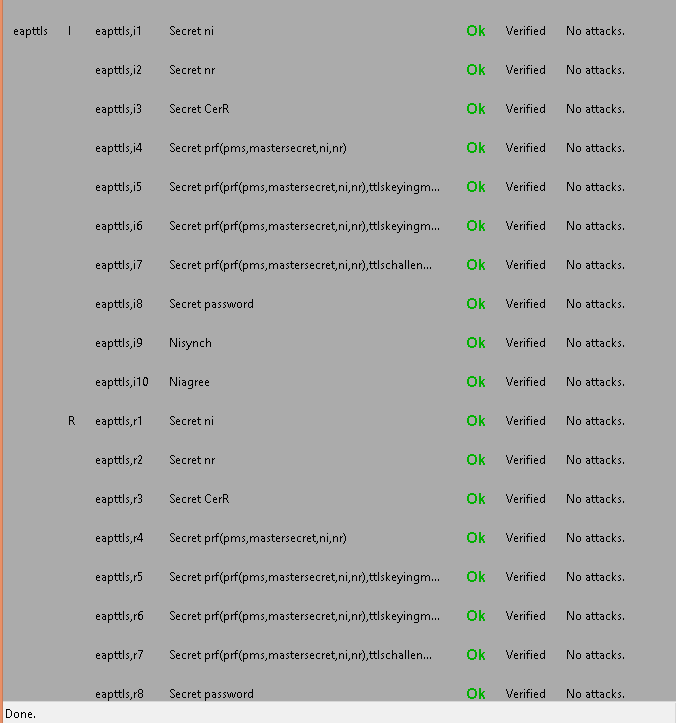
const CerR: Data;

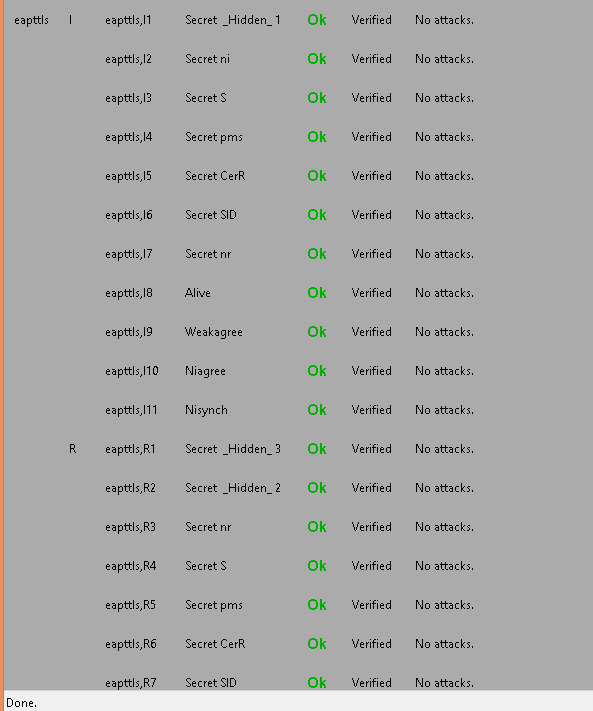
recv\_2 (R, CA, R);

send\_3 (CA, R, {R, {CerR, pk(R)}pk(R)}sk(CA));

}

}





EAP-TTLS modified

hashfunction prf;

hashfunction h;

usertype String;

const ccs, csu, success, accesslevel, clientfinished, serverfinished: String;

const ttlschallenge, ttlskeyingmaterial,mastersecret, shd, password, uname, unamenottrue: String;

macro ms= prf (pms, mastersecret, (ni, nr));

macro finishedi= h(ms, clientfinished, SID, ni, csu, nr, {CerR, pk(R)}sk(CA), shd, {pms}pk(R));

macro finisheds= h(ms, serverfinished, SID, ni, csu, nr, {CerR, pk(R)}sk(CA), shd, {pms}pk(R) , finishedi);

macro clientkey= prf (ms, ttlskeyingmaterial, (ni, nr));

macro serverkey= prf (ms, ttlskeyingmaterial, (ni, nr));

macro chapchallenge= h (ms, ttlschallenge, ni,nr);

macro mschap= (uname, chapchallenge, password);

protocol eapttls (I, R, CA)

{

role I

{

const SID, CerR, pms, S: Data;

fresh ni: Nonce;

var nr: Nonce;

send\_1 (I, R, unamenottrue, SID, ni, csu);

recv\_4 (R, I, SID, nr, csu, {CerR, pk(R)}sk(CA), shd);

send\_5 (I, R, {pms}pk(R), ccs);

send\_6 (I, R, finishedi);

recv\_7 (R, I, ccs, finisheds);

match (h (ms, serverfinished, SID, ni, csu, nr, {CerR, pk(R)}sk(CA), shd, {pms}pk(R) , finishedi), finisheds);

send\_9 (I, R, {mschap}clientkey);

recv\_10 (R, I, {success, accesslevel} serverkey);

claim\_i1 (I, Secret, ni);

claim\_i2 (I, Secret, nr);

claim\_i3 (I, Secret, CerR);

claim\_i4 (I, Secret, ms);

claim\_i5 (I, Secret, clientkey);

claim\_i6 (I, Secret, serverkey);

claim\_i7 (I, Secret, chapchallenge);

claim\_i8 (I, Secret, password);

claim\_i9 (I, Nisynch);

claim\_i10 (I, Niagree);

}

role R

{

const SID, CerR, pms, S: Data;

fresh nr: Nonce;

var ni: Nonce;

recv\_1 (I, R, unamenottrue, SID, ni, csu);

send\_2 (R, CA, R);

recv\_3 (CA, R, {R, {CerR, pk(R)}pk(R)}sk(CA));

send\_4 (R, I, SID, nr, csu, {CerR, pk(R)}sk(CA), shd);

recv\_5(I, R, {pms}pk(R), ccs);

recv\_6 (I, R, finishedi);

match (h (ms, clientfinished, SID, ni, csu, nr, {CerR, pk(R)}sk(CA), shd, {pms}pk(R)), finishedi);

send\_7(R, I, ccs, finisheds);

recv\_9 (I, R, {mschap}clientkey);

match ((uname, h (ms, ttlschallenge, ni,nr), password), mschap);

send\_10 (R, I, {success, accesslevel}serverkey);

claim\_r1 (R, Secret, ni);

claim\_r2 (R, Secret, nr);

claim\_r3 (R, Secret, CerR);

claim\_r4 (R, Secret, ms);

claim\_r5 (R, Secret, clientkey);

claim\_r6 (R, Secret, serverkey);

claim\_r7 (R, Secret, chapchallenge);

claim\_r8 (R, Secret, password);

claim\_r9 (R, Nisynch);

claim\_r10 (R, Niagree);

}

role CA

{

const CerR: Data;

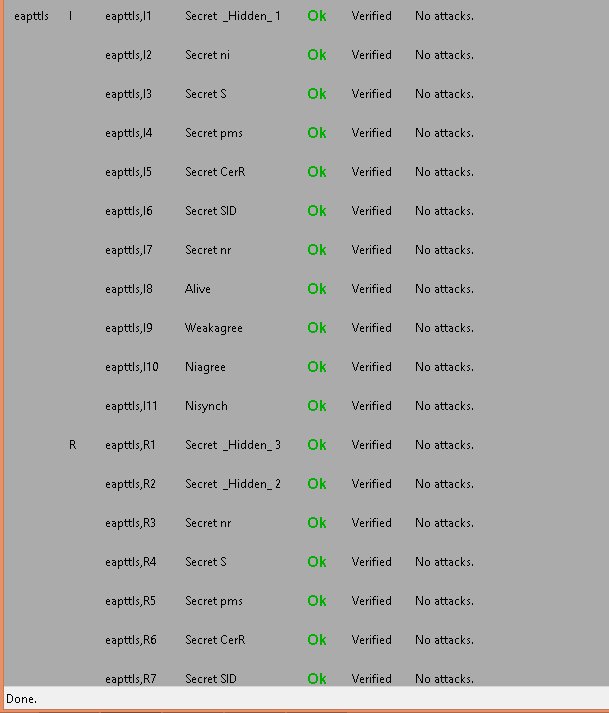
recv\_2 (R, CA, R);

send\_3 (CA, R, {R, {CerR, pk(R)}pk(R)}sk(CA));

}

}





MATLAB simulation of EAP-TTLS

clc;

clear;

trsae= 0.1667 %rsaencryption

trsad= 0.1667 %rsadecryption

thmacsha256= 0.0714

tsha1= 0.0384; %sha

tsha256= 0.05 %sha256

tdese= 0.29165; %desencryption

tdesd= 0.29165; %desdecryption

tmd4= 0.0476

%EAP-TTLS AUTHENTICATION

eapttlso\_a = [trsae, 5\*thmacsha256+2\*thmacsha256, 2\*thmacsha256,14\*thmacsha256+2\*thmacsha256+tdese, tdesd];

eapttlso\_b= [trsad,5\*thmacsha256+2\*thmacsha256, 2\*thmacsha256,14\*thmacsha256+2\*thmacsha256+tdesd, tdese];

eapttlso=[ trsae, trsad, 5\*thmacsha256+2\*thmacsha256, 5\*thmacsha256+2\*thmacsha256, 2\*thmacsha256, 2\*thmacsha256, 14\*thmacsha256+2\*thmacsha256+tdese, 14\*thmacsha256+2\*thmacsha256+tdesd,tdese,tdesd ];

%total===============================

for tmp = 2:length(eapttlso\_a);

eapttlso\_a (tmp)= eapttlso\_a (tmp-1)+ eapttlso\_a (tmp);

end

for tmp = 2:length(eapttlso);

eapttlso (tmp)= eapttlso (tmp-1)+ eapttlso (tmp);

end

%EAP-TLS AUTHENTICATION Modified

eapttlsm\_a = [trsae, 2\*thmacsha256+tsha1+thmacsha256,thmacsha256,4\*thmacsha256+6\*tsha1+tmd4+tdese, tdesd];

eapttlsm\_b= [trsad,2\*thmacsha256+tsha1+thmacsha256, thmacsha256, 4\*thmacsha256+tsha1+tmd4+tdesd, tdese];

eapttlsm=[trsae, trsad, 2\*thmacsha256+tsha1+thmacsha256, 2\*thmacsha256+tsha1+thmacsha256, thmacsha256, thmacsha256, 4\*thmacsha256+6\*tsha1+tmd4+tdese, 4\*thmacsha256+tsha1+tmd4+tdesd, tdese, tdesd];

%total===============================

for tmp = 2:length(eapttlsm\_a);

eapttlsm\_a (tmp)= eapttlsm\_a (tmp-1)+ eapttlsm\_a (tmp);

end

for tmp = 2:length(eapttlsm);

eapttlsm (tmp)= eapttlsm (tmp-1)+ eapttlsm (tmp);

end

total\_number = 100000;

unkown\_attacks= 0;

y\_eapttlso = zeros(1,10);

y\_eapttlsm= zeros(1,10);

left\_time\_eapttlso= 0;

left\_time\_eapttlsm=0;

n=1;

for x=0:0.1:0.9

left\_time\_eapttlso= total\_number\*(1-x)\*eapttlso\_a(length(eapttlso\_a));

left\_time\_eapttlsm= total\_number\*(1-x)\*eapttlsm\_a(length(eapttlsm\_a));

unknown\_attacks = uint16(total\_number\*x);

unexpected\_delay\_eapttlso = randi([1,length(eapttlso\_a)],1,unknown\_attacks);

unexpected\_delay\_eapttlsm = randi([1,length(eapttlsm\_a)],1,unknown\_attacks);

attack\_total\_delay\_eapttlso= 0;

attack\_total\_delay\_eapttlsm= 0;

for i=1:unknown\_attacks

attack\_total\_delay\_eapttlso = attack\_total\_delay\_eapttlso + eapttlso\_a(unexpected\_delay\_eapttlso(i));

attack\_total\_delay\_eapttlsm=attack\_total\_delay\_eapttlsm+eapttlsm\_a(unexpected\_delay\_eapttlsm(i));

end

y\_eapttlso(n)=(left\_time\_eapttlso+attack\_total\_delay\_eapttlso)/(total\_number\*(1-x));

y\_eapttlsm(n)=(left\_time\_eapttlsm+attack\_total\_delay\_eapttlsm)/(total\_number\*(1-x));

n=n+1;

end

x=0:0.1:0.9;

%figure;

plot(x,y\_eapttlso,'-k', x, y\_eapttlsm,'-.k');

set(gca,'XTick',0:0.1:1);

set(gca, 'xticklabel', {'0','0.1','0.2','0.3','0.4','0.5','0.6','0.7','0.8','0.9','1'});

xlabel('Ratio of unknown to known attacks','fontsize',12);

ylabel('Total Time Delay (ms)','fontsize',12);

legend({'EAP-TTLSo', 'EAP-TTLSm'},'FontSize',12,'FontWeight','bold');

axis([0,0.9,0,10]);

