Mschapv2 original

hashfunction h1;

hashfunction h2;

usertype string;

const p,x, clientsuccess, serversuccess, accesslevel: string;

const magicservertoclientsigningconstant, padtomakeitdomorethanoneiteration: string;

macro m1= h1(ni, nr, x);

macro m2= h2(p);

macro m3= {m1}m2;

macro m4= h2(h2(p));

macro m5= magicservertoclientsigningconstant;

macro m6= h1(m4,m3,m5);

macro m7= padtomakeitdomorethanoneiteration;

macro m8= h1(m6, m1,m7);

protocol mschap(I,R)

{

role I

{

fresh ni: Nonce;

var nr: Nonce;

recv\_1 (R, I, nr);

send\_2 (I, R, ni, x, m3);

recv\_3 (R, I, {clientsuccess}pk(I));

recv\_4 (R, I, m8);

match (h1(h1(h2(h2(p)), m3, m5), m1,m7), m8);

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

recv\_6 (R, I ,{accesslevel}pk(I));

}

role R

{

fresh nr: Nonce;

var ni: Nonce;

send\_1 (R, I, nr);

recv\_2 (I, R, ni, x, m3);

match (({h1(ni, nr, x)}h2(p)), m3);

send\_3 (R, I, {clientsuccess}pk(I));

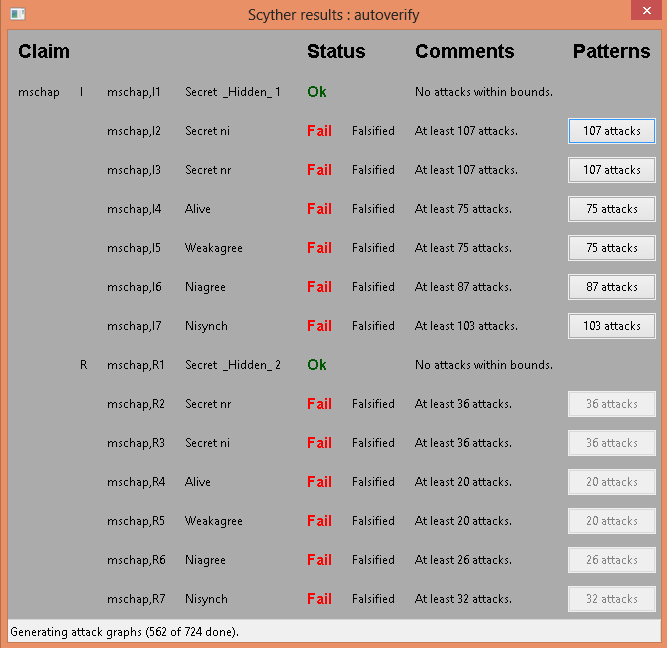
send\_4 (R, I, m8);

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

send\_6 (R, I ,{accesslevel}pk(I));

}

}



MS-CHAP V2 modified

hashfunction h1;

hashfunction h2;

hashfunction h3;

usertype string;

const p,x, clientsuccess, serversuccess, accesslevel: string;

const magicservertoclientsigningconstant, padtomakeitdomorethanoneiteration: string;

macro m1= h1(ni, nr, x);

macro m2= h2(p);

macro me= {h2(p)}nr;

macro m3= {m1}me;

macro m4= h3(m1,m2,m3);

protocol mschap(I,R)

{

role I

{

fresh ni: Nonce;

var nr: Nonce;

recv\_1 (R, I, {{nr}pk(I)}sk(R));

send\_2 (I, R, {{ni, x}pk(R)}sk(I), m3);

recv\_3 (R, I, {{clientsuccess}pk(I)}sk(R));

recv\_4 (R, I, m4);

match (h1(h1 (ni, nr, x),h2(p), m3), m4);

send\_5 (I, R, {{serversuccess}pk(R)}sk(I));

recv\_6 (R, I ,{{accesslevel}pk(I)}sk(R));

claim\_i1 (I, Secret, ni);

claim\_i2 (I, Secret, nr);

claim\_i3 (I, Secret, p);

claim\_i4 (I, Secret, m1);

claim\_i5 (I, Secret, m2);

claim\_i6 (I, Niagree);

claim\_i7 (I, Nisynch);

}

role R

{

fresh nr: Nonce;

var ni: Nonce;

send\_1 (R, I, {{nr}pk(I)}sk(R));

recv\_2 (I, R, {{ni, x}pk(R)}sk(I), m3);

match (({h1(ni, nr, x)}({h2(p)}nr)), m3);

send\_3 (R, I, {{clientsuccess}pk(I)}sk(R));

send\_4 (R, I, m4);

recv\_5 (I, R, {{serversuccess}pk(R)}sk(I));

send\_6 (R, I ,{{accesslevel}pk(I)}sk(R));

claim\_r1 (R, Secret, ni);

claim\_r2 (R, Secret, nr);

claim\_r3 (R, Secret, p);

claim\_r4 (R, Secret, m1);

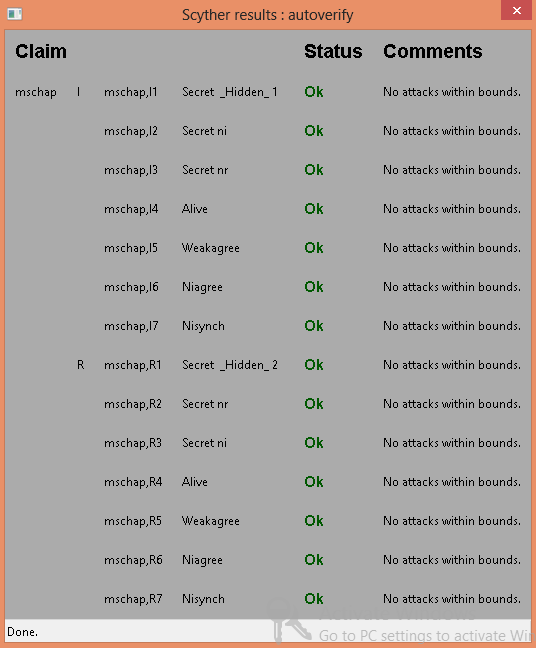
claim\_r5 (R, Secret, m2);

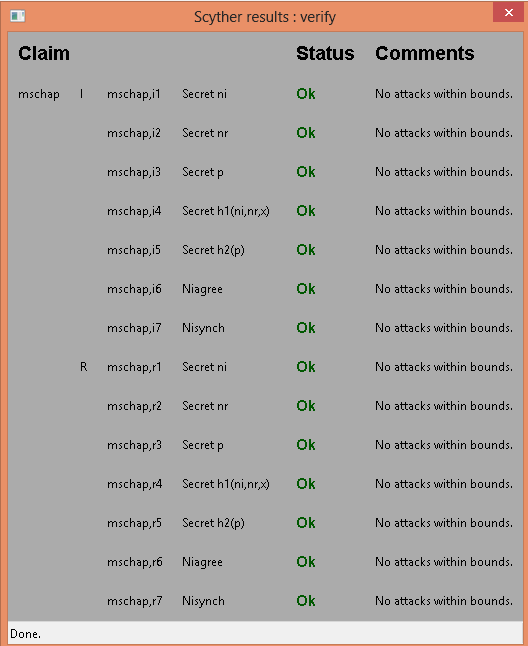
claim\_r6 (R, Niagree);

claim\_r7 (R, Nisynch);

}

}





Assuming continuous calculation of hashes:

clc;

clear;

tsha1= 0.0384; %sha1

tmd4= 0.0476; %md4

tsha256= 0.05;

tdese= 0.29165 %desencryption

tdesd= 0.29165 %desdecryption

trsae= 0.14285 %rsaencryption

trsad= 0.14285 %trsadecryption

%MS-CHAPV2 AUTHENTICATION

CHAPo\_a = [tsha1+tmd4+3\*tdese, trsad, tmd4+tsha1+tsha1, trsae, trsad];

CHAPo\_b= [tsha1+tmd4+3\*tdese, trsae, tmd4+tsha1+tsha1, trsad, trsae];

CHAPo=[tsha1+tmd4+3\*tdese, tsha1+tmd4+3\*tdese +3\*tdesd, trsae, trsad tmd4+tsha1+tsha1, tmd4+tsha1+tsha1, trsae, trsad, trsae, trsad];

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

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

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

end

for tmp = 2:length(CHAPo);

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

end

%MS-CHAPV2 AUTHENTICATION Modified

CHAPm\_a = [2\*trsad, 2\*trsae+tsha1+tmd4+tdese+3\*tdese, 2\*trsad, tsha256, 2\*trsae, 2\*trsad];

CHAPm\_b= [2\*trsae, 2\*trsad+tsha1+tmd4+tdese+3\*tdese, 2\*trsae, tsha256,2\*trsad, 2\*trsae];

CHAPm=[2\*trsae,2\*trsad, 2\*trsae+tsha1+tmd4+tdese+3\*tdese, 2\*trsad+tsha1+tmd4+tdese+3\*tdese, 2\*trsae, 2\*trsad, tsha256, tsha256,2\*trsae,2\*trsad, 2\*trsae, 2\*trsad ]

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

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

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

end

for tmp = 2:length(CHAPm);

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

end

total\_number = 100000;

unkown\_attacks= 0;

y\_mschapo = zeros(1,10);

y\_mschapm= zeros(1,10);

left\_time\_mschapo= 0;

left\_time\_mschapm=0;

n=1;

for x=0:0.1:0.9

left\_time\_mschapo= total\_number\*(1-x)\*CHAPo\_a(length(CHAPo\_a));

left\_time\_mschapm= total\_number\*(1-x)\*CHAPm\_a(length(CHAPm\_a));

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

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

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

attack\_total\_delay\_mschapo= 0;

attack\_total\_delay\_mschapm= 0;

for i=1:unknown\_attacks

attack\_total\_delay\_mschapo = attack\_total\_delay\_mschapo + CHAPo\_a(unexpected\_delay\_mschapo(i));

attack\_total\_delay\_mschapm=attack\_total\_delay\_mschapm+CHAPm\_a(unexpected\_delay\_mschapm(i));

end

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

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

n=n+1;

end

x=0:0.1:0.9;

%figure;

plot(x,y\_mschapo,'-k', x, y\_mschapm,'-.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'});

% title('Delay','fontsize',12);

xlabel('Ratio of Unknown to Known attacks','fontsize',12);

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

legend({'MSCHAPo', 'MSCHAPm'},'FontSize',12,'FontWeight','bold');

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

