



Verificação Automática de Protocolos de Segurança com a ferramenta Scyther

Tadeu Jenuario, João Otávio Chervinski, Giulliano Paz, Diego Kreutz

4o. Workshop Regional de Segurança da Informação (2019)



Key Reinstallation Attacks

Breaking WPA2 by forcing nonce reuse

Discovered by [Mathy Vanhoef](#) of [imec-DistriNet](#), KU Leuven, 2017

Dragonblood: Data-leaking flaw in
WPA3 Wi-Fi authentication

TLS Security 6: Examples of TLS Vulnerabilities and Attacks

POSTED ON MARCH 31, 2019 BY AGATHOKLIS PRODROMOU

Alert (TA14-290A)

[More Alerts](#)

SSL 3.0 Protocol Vulnerability and POODLE Attack

Original release date: October 17, 2014 | Last revised: September 30, 2016

Verificação Formal

Ferramenta de Verificação

Ferramenta Scyther

Considerações Finais

O Que é Verificação Formal?

O que é Verificação Formal?

- Verificar um protocolo é demonstrar sua corretude.
- Por que utilizar a Verificação Formal?
 - Garantir que o protocolo atinja o objetivo
 - Descobrir vulnerabilidades
 - Incitar interesse comercial
 - Precipitar movimentos de invasores
- Existem vários modos de se fazer a verificação formal

Verificação Formal

Ferramentas de Verificação

Ferramenta Scyther

Considerações Finais

Ferramentas de Verificação

- Avispa
- ProVerif
- Tamarin
- Scyther

Verificação Formal

Ferramentas de Verificação

Ferramenta Scyther

Considerações Finais

Ferramenta Scyther

Semânticas Operacionais

Verificação Formal de Protocolos

Semânticas Operacionais

- Termos atômicos:
 - fresh
 - var
 - const
- Chaves assimétricas:
 - $sk(X)$
 - $pk(X)$

Semânticas Operacionais

- Tipos predefinidos:
 - Agent
 - Function
 - Nonce
- Tipos básicos de eventos:
 - send
 - recv
- Eventos de afirmação:
 - secret
 - Nisynch

Ferramenta Scyther

Semânticas Operacionais

Verificação Formal de Protocolos

Verificação Formal do Protocolo

Protocolo de Needham-Schroeder

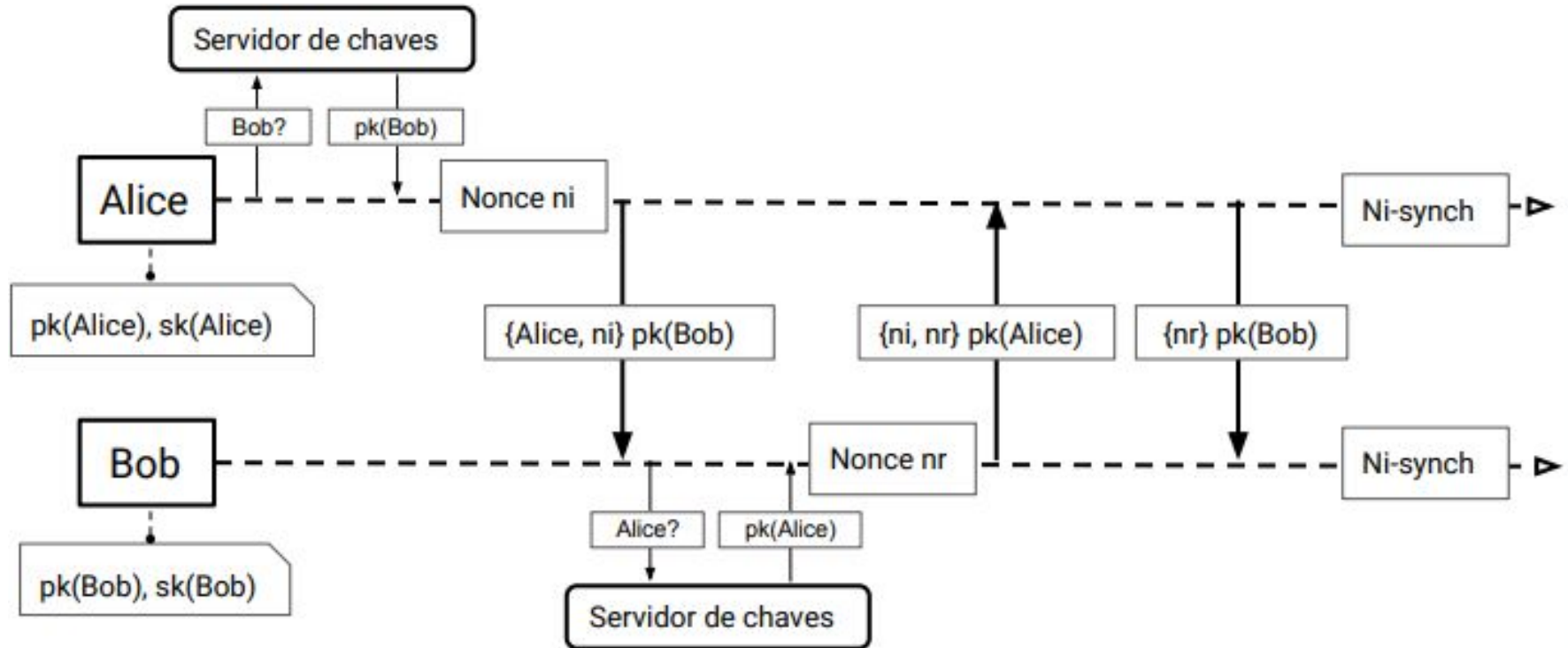
Algoritmo na Semântica da Scyther

Análise do Protocolo com a Ferramenta Scyther

Protocolo de Needham-Schroeder

- Características:
 - Método de autenticação entre dois participantes
 - Rede insegura
 - Apresenta falhas
 - Utiliza criptografia assimétrica

Protocolo de Needham-Schroeder



Verificação Formal do Protocolo

Protocolo de Needham-Schroeder

Algoritmo na Semântica da Scyther

Análise do Protocolo com a Ferramenta Scyther

Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);  
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```

Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);  
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```

Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);  
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```


Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);  
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```

Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);  
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```

Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);  
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```


Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);  
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```


Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);  
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```

Algoritmo na Semântica da Scyther

```
1  const pk: Function;  
2  secret sk: Function;  
3  inversekeys (pk,sk);  
4  const Eve: Agent;  
5  untrusted Eve;  
6  protocol ns(Alice,Bob,Eve){  
7    role Alice{  
8      fresh ni: Nonce;  
9      var nr: Nonce;  
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));  
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));  
12     send_3(Alice,Bob,{ nr }pk(Bob));  
13     claim(Alice,Secret,ni);  
14     claim(Alice,Secret,nr);
```

```
15     claim(Alice,Nisynch);  
16   }  
17   role Bob{  
18     var ni: Nonce;  
19     fresh nr: Nonce;  
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));  
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));  
22     recv_3(Alice,Bob,{ nr }pk(Bob));  
23     claim(Bob,Secret,ni);  
24     claim(Bob,Secret,nr);  
25     claim(Bob,Nisynch);  
26   }  
27 }
```

Verificação Formal do Protocolo

Protocolo de Needham-Schroeder

Algoritmo na Semântica da Scyther

Análise do Protocolo com a Ferramenta Scyther

Análise do Protocolo com a Ferramenta Scyther

Scyther: needham_schroeder.spdl

File Verify Help

Protocol description Settings

```
1
2 const pk: Function;
3 secret sk: Function;
4 inversekeys (pk,sk);
5 const Eve: Agent;
6 untrusted Eve;
7
8 //The protocol description
9
10 protocol ns3(Alice,Bob,Eve)
11 {
12     role Alice
13     {
14         fresh ni: Nonce;
15         var nr: Nonce;
16
17         send_1(Alice,Bob,{Alice,ni}pk(Bob));
18         recv_2(Bob,Alice,{ni,nr}pk(Alice));
19         send_3(Alice,Bob,{nr}pk(Bob));
20     }
```

Scyther: needham_schroeder.spdl

File Verify Help

Prot

Verify protocol	F1
Characterize roles	F2
Verify automatic claims	F6

```
1
2
3
4 inversekeys (pk,sk);
5 const Eve: Agent;
6 untrusted Eve;
7
8 //The protocol description
9
10 protocol ns3(Alice,Bob,Eve)
11 {
12     role Alice
13     {
14         fresh ni: Nonce;
15         var nr: Nonce;
16
17         send_1(Alice,Bob,{Alice,ni}pk(Bob));
18         recv_2(Bob,Alice,{ni,nr}pk(Alice));
19         send_3(Alice,Bob,{nr}pk(Bob));
20     }
```

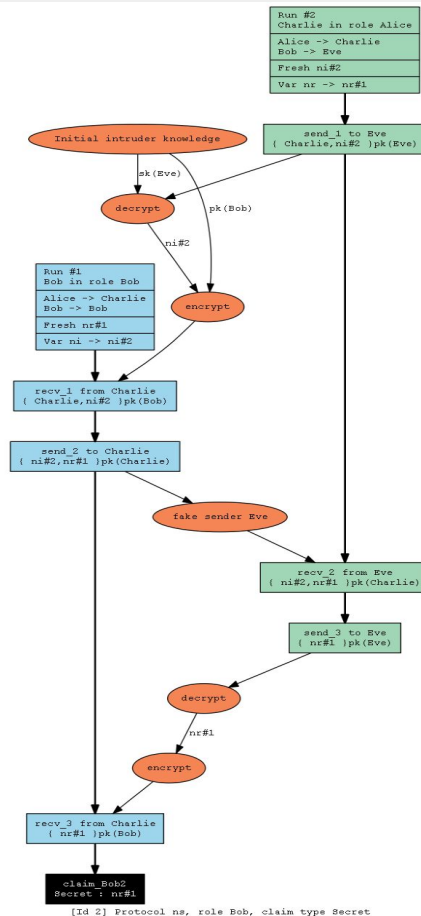
Análise do Protocolo com a Ferramenta Scyther

Scyther results : verify

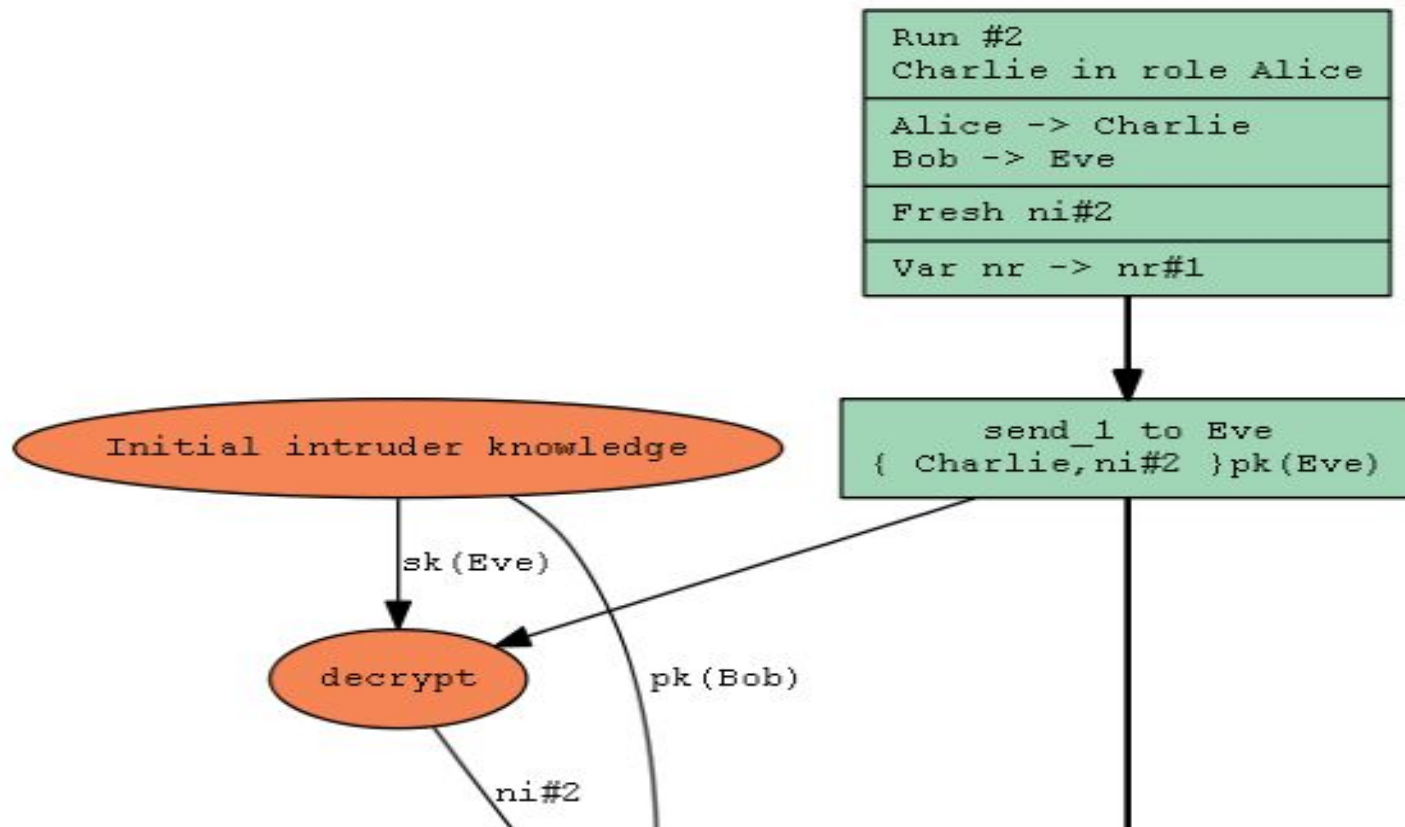
Claim				Status	Comments	Patterns
ns3	Alice	ns3,Alice1	Secret ni	Ok	Verified	No attacks.
		ns3,Alice2	Secret nr	Ok	Verified	No attacks.
		ns3,Alice3	Nisynch	Ok	Verified	No attacks.
Bob		ns3,Bob1	Secret ni	Fail	Falsified	At least 1 attack. <div>1 attack</div>
		ns3,Bob2	Secret nr	Fail	Falsified	At least 1 attack. <div>1 attack</div>
		ns3,Bob3	Nisynch	Fail	Falsified	At least 1 attack. <div>1 attack</div>

Done.

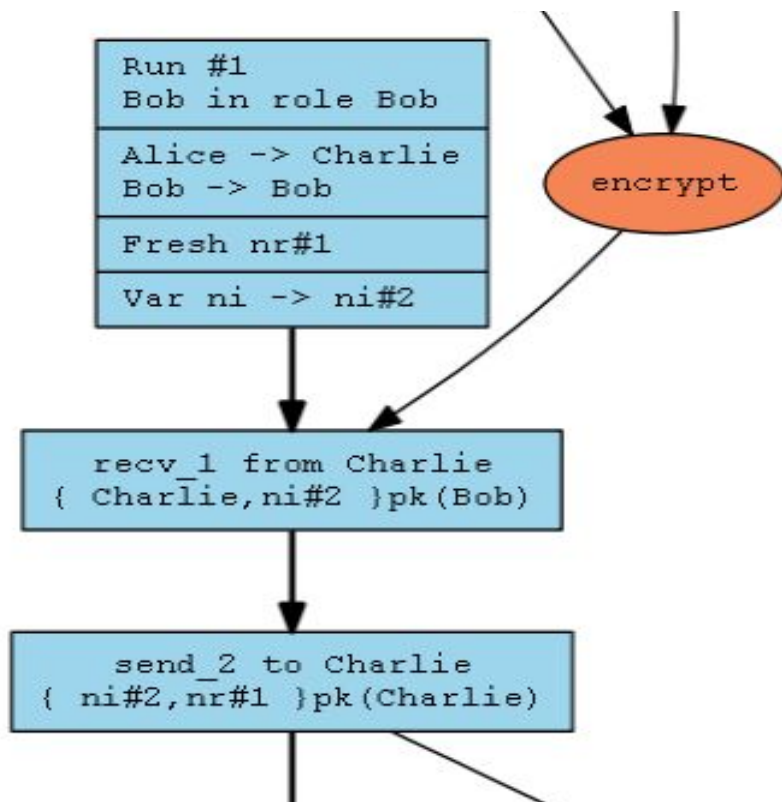
Análise do Protocolo com a Ferramenta Scyther



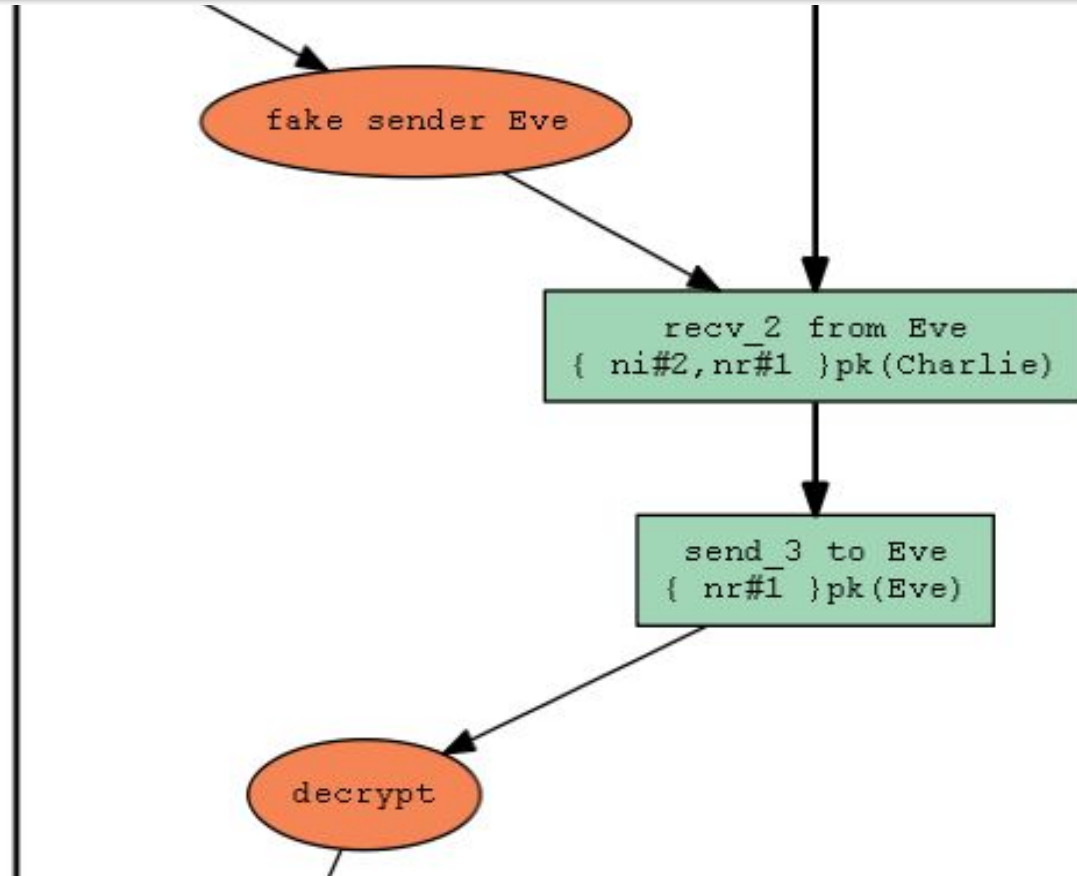
Análise do Protocolo com a Ferramenta Scyther



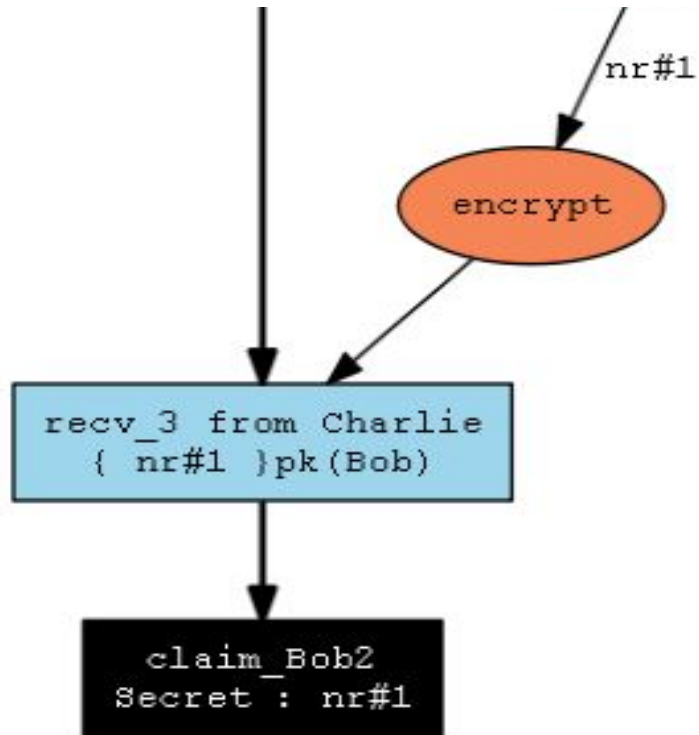
Análise do Protocolo com a Ferramenta Scyther



Análise do Protocolo com a Ferramenta Scyther



Análise do Protocolo com a Ferramenta Scyther



[Id 2] Protocol ns, role Bob, claim type Secret

Análise do Protocolo com a Ferramenta Scyther

```
1  const pk: Function;
2  secret sk: Function;
3  inversekeys (pk,sk);
4  const Eve: Agent;
5  untrusted Eve;
6  protocol ns(Alice,Bob,Eve){
7    role Alice{
8      fresh ni: Nonce;
9      var nr: Nonce;
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));
11     recv_2(Bob,Alice,{ ni,nr }pk(Alice));
12     send_3(Alice,Bob,{ nr }pk(Bob));
13     claim(Alice,Secret,ni);
14     claim(Alice,Secret,nr);
15     claim(Alice,Nisynch);
16   }
17   role Bob{
18     var ni: Nonce;
19     fresh nr: Nonce;
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));
21     send_2(Bob,Alice,{ ni,nr }pk(Alice));
22     recv_3(Alice,Bob,{ nr }pk(Bob));
23     claim(Bob,Secret,ni);
24     claim(Bob,Secret,nr);
25     claim(Bob,Nisynch);
26   }
27 }
```

Análise do Protocolo com a Ferramenta Scyther

```
1  const pk: Function;
2  secret sk: Function;
3  inversekeys (pk,sk);
4  const Eve: Agent;
5  untrusted Eve;
6  protocol ns(Alice,Bob,Eve){
7    role Alice{
8      fresh ni: Nonce;
9      var nr: Nonce;
10     send_1(Alice,Bob,{ Alice,ni }pk(Bob));
11     recv_2(Bob,Alice,{ Bob,ni,nr }pk(Alice));
12     send_3(Alice,Bob,{ nr }pk(Bob));
13     claim(Alice,Secret,ni);
14     claim(Alice,Secret,nr);
15     claim(Alice,Nisynch);
16   }
17   role Bob{
18     var ni: Nonce;
19     fresh nr: Nonce;
20     recv_1(Alice,Bob,{ Alice,ni }pk(Bob));
21     send_2(Bob,Alice,{ Bob,ni,nr }pk(Alice));
22     recv_3(Alice,Bob,{ nr }pk(Bob));
23     claim(Bob,Secret,ni);
24     claim(Bob,Secret,nr);
25     claim(Bob,Nisynch);
26   }
27 }
```

Análise do Protocolo com a Ferramenta Scyther



The image shows a screenshot of a software window titled "Scyther results : verify". The window contains a table with three columns: "Claim", "Status", and "Comments". The table lists six rows of verification results, grouped by participants Alice and Bob. Each row shows a specific claim (e.g., "Secret ni", "Secret nr", "Nisynch") and its status ("Ok", "Verified", "No attacks"). The status "Ok" is highlighted in green. At the bottom of the window, there is a status bar that says "Done."

Claim				Status		Comments
ns3	Alice	ns3,Alice1	Secret ni	Ok	Verified	No attacks.
		ns3,Alice2	Secret nr	Ok	Verified	No attacks.
		ns3,Alice3	Nisynch	Ok	Verified	No attacks.
	Bob	ns3,Bob1	Secret ni	Ok	Verified	No attacks.
		ns3,Bob2	Secret nr	Ok	Verified	No attacks.
		ns3,Bob3	Nisynch	Ok	Verified	No attacks.

Done.

Análise do Protocolo com a Ferramenta Scyther

```
1  const pk: Function;
2  secret sk: Function;
3  inversekeys (pk,sk);
4  const Eve: Agent;
5  untrusted Eve;
6  protocol ns(Alice,Bob,Eve){
7    role Alice{
8      fresh ni: Nonce;
9      var nr: Nonce;
10     send_1(Alice,Bob,{ Alice,ni}pk(Bob));
11     recv_2(Bob,Alice,{ ni,nr}pk(Alice));
12     send_3(Alice,Bob,{ nr}pk(Bob));
13     compromised sk(Eve);
14     claim(Alice,Secret,ni);
```

```
15     claim(Alice,Secret,nr);
16     claim(Alice,Nisynch);
17   }
18   role Bob{
19     var ni: Nonce;
20     fresh nr: Nonce;
21     recv_1(Alice,Bob,{ Alice,ni}pk(Bob));
22     send_2(Bob,Alice,{ ni,nr}pk(Alice));
23     recv_3(Alice,Bob,{ nr}pk(Bob));
24     claim(Bob,Secret,ni);
25     claim(Bob,Secret,nr);
26     claim(Bob,Nisynch);
27   }
28 }
```


Análise do Protocolo com a Ferramenta Scyther

Claim				Status	Comments	Patterns
ns3 Alice	ns3,Alice1	Secret ni	Fail	Falsified	At least 1 attack.	1 attack
	ns3,Alice2	Secret nr	Fail	Falsified	At least 1 attack.	1 attack
	ns3,Alice3	Nisynch	Fail	Falsified	At least 1 attack.	1 attack
Bob	ns3,Bob1	Secret ni	Fail	Falsified	At least 1 attack.	1 attack
	ns3,Bob2	Secret nr	Fail	Falsified	At least 1 attack.	1 attack
	ns3,Bob3	Nisynch	Fail	Falsified	At least 1 attack.	1 attack

Done.

Verificação Formal

Ferramentas de Verificação

Ferramenta Scyther

Considerações Finais

Considerações finais

Detecção de Falhas

Corretude do Protocolo

Aplicação Prática

Automatização

Obrigado!

tadeujenuario@hotmail.com

joaootaviors@gmail.com

giulliano94@gmail.com

kreutz@acm.org