Skype uncovered

Security study of Skype

Desclaux Fabrice[1]

[1]EADS CCR/STI/C

EADS CCR

Quick overview of Skype

End-user view

- Perfect VoIP software with good quality sound
- Ease of use and working everywhere and with every OS

Network administrator view

- Skype bypasses Firewalls, Nat, Proxies
- It uses P2P technologies
- Skype traffic cannot be isolated and is suspicious
- In a nutshell, the perfect backdoor

EADS CCR

Why is Skype seen so suspicious ?

The Binary

- Big size (about 12Mo)
- *strings* doesn't reveal interesting things
- Few functions in the binary import table
- The binary doesn't want to launch if the *Soft-ice* debugger is present

The network

- Protocol is proprietary and not obvious to observe
- The number of boxes contacted by a client is very important

Conclusion

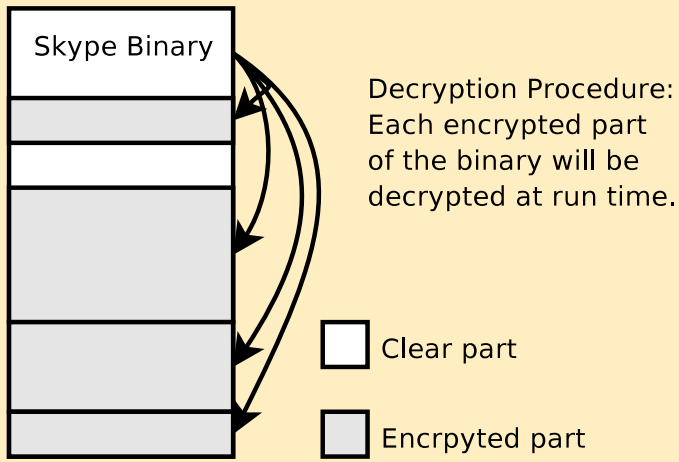
$\Longrightarrow$ Skype is a total black box.

EADS CCR

Binary analysis : Encryption

Encryption layers

- Some parts of the binary are *xored* by a hard-coded key in the code
- In memory, Skype is fully decrypted



Binary protection : Anti debuggers

Anti Softice

- Some tests are done in order to detect the Softice debugger
- First tests are easy to detect
- The others are hidden in the binary

Binary protection : Anti debuggers

Example

First Softice test

```
mov eax, offset str_Siwvid ; "\\\\.\\Siwvid"
call test_driver
test al, al
```

Example

Hidden test : It checks if Softice is not in the Driver list.

```
call EnumDeviceDrivers
...
call GetDeviceDriverBaseNameA
...
cmp eax, 'ntic'
jnz next_
cmp ebx, 'e.sy'
jnz next_
cmp ecx, 's\x00\x00\x00'
jnz next_
```

Binary protection : Import functions

Hidden imports

- In a common binary, imported libraries and functions are described in a structure
- In Skype only some functions are present
- The other part is dynamically loaded after decryption
- $\Longrightarrow$This prevent disassemblers from watching interesting functions

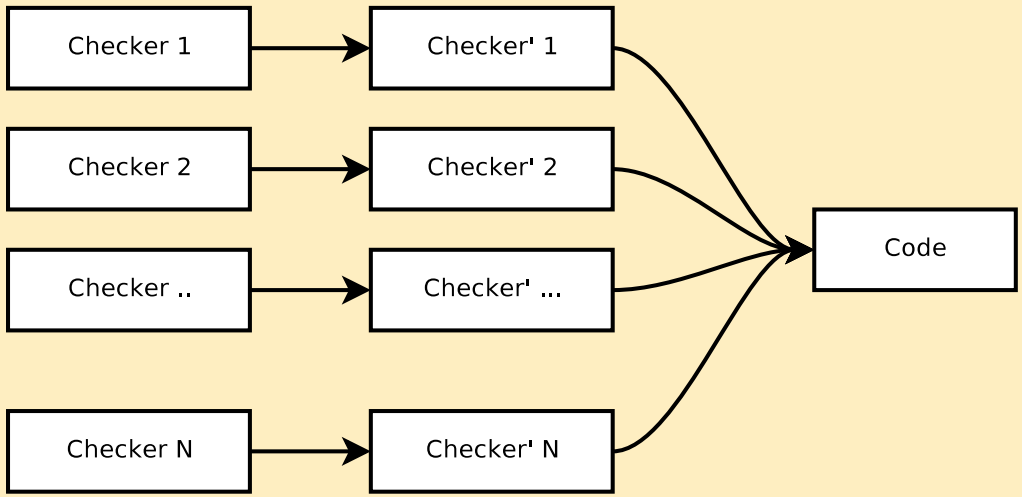
Example

Libraries used in hidden imports :	Number of total hidden imports :
KERNEL32.dll WINMM.dll WS2_32.dll RPCRT4.dll ...	169/843

Binary analysis : Integrity

Multiple checksums

Skype checks its own integrity by implementing thousands of code checkers. If a software breakpoint is installed, or a modification is done in the binary, Skype will stop/crash randomly.



Main scheme of Skype code checkers

Binary analysis : Obfuscation

Code obfuscation

- Some parts of the binary are obfuscated. This may be used in order to avoid *Skype light remakes*
- The next code represents a code checker that is generated to avoid being detected by IDA
- Pointers are calculated, junk code is inserted in the real code

```
start:
    xor     edi, edi
    add     edi, 0x688E5C
    mov     eax, 0x320E83
    xor     eax, 0x1C4C4
    mov     ebx, eax
    add     ebx, 0xFFCC5AFD
loop_start:
    mov     ecx, [edi+0x10]
    jmp     lbl1
    db 0x19
lbl1:
    sub     eax, ecx
    sub     edi, 1
    dec     ebx
    jnz     loop_start
    jmp     lbl2
    db 0x73
lbl2:
    jmp     lbl3
    dd 0xC8528417, 0xD8FBBD1, 0xA36CFB2F, 0xE8D6E4B7, 0xC0B8797A
    db 0x61, 0xBD
lbl3:
    sub     eax, 0x4C49F346
```

EADS CCR

Protocol analysis

Indication Packets

Most packets are compounded in two parts :

- A clear header
- A ciphered payload. The payload is ciphered with a RC4 stream

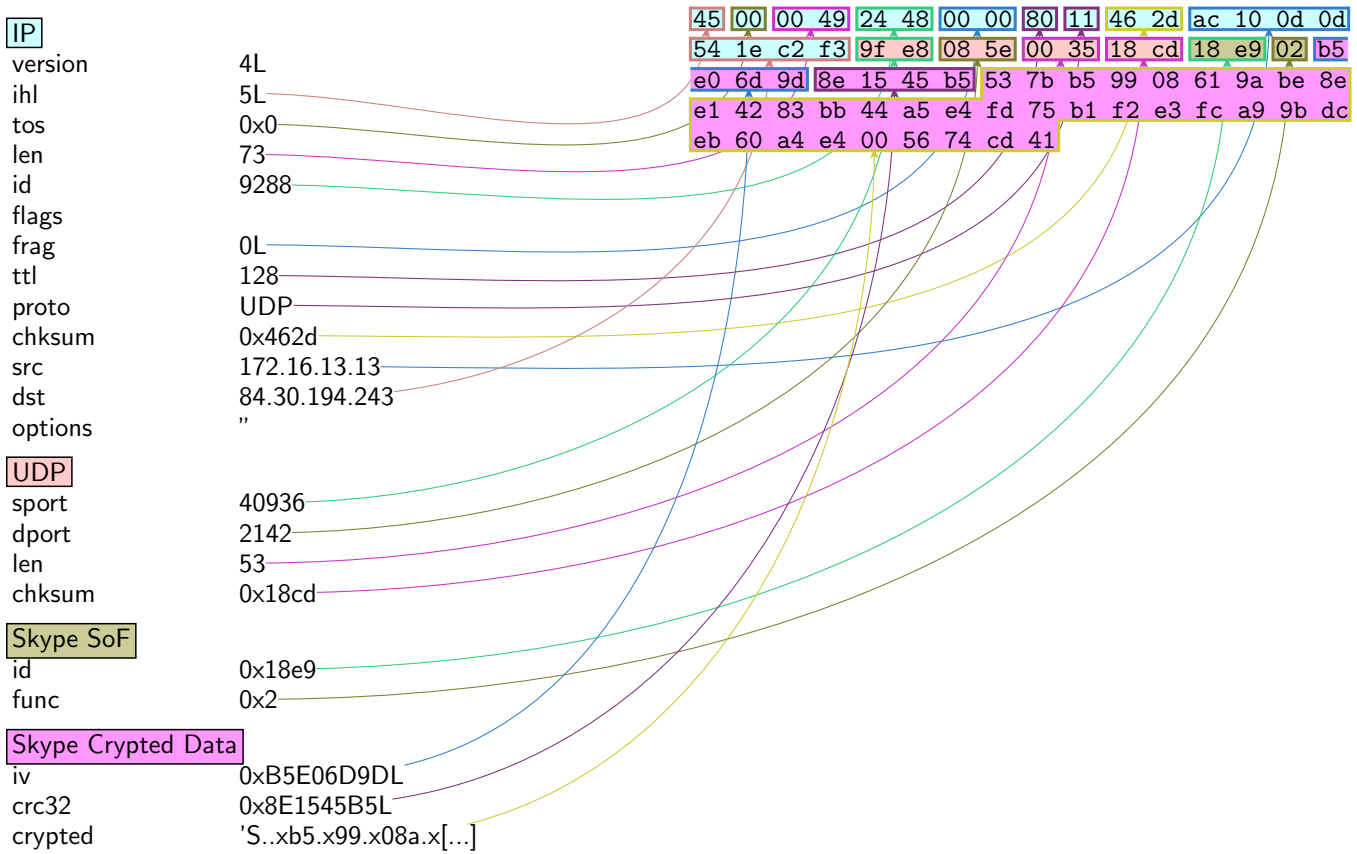
Signalling Packets

- The RC4 is only used to obfuscate the packet payload
- That's why a simple *tcpdump* doesn't reveal interesting things
- RC4 key can be recovered from the packet (UDP)

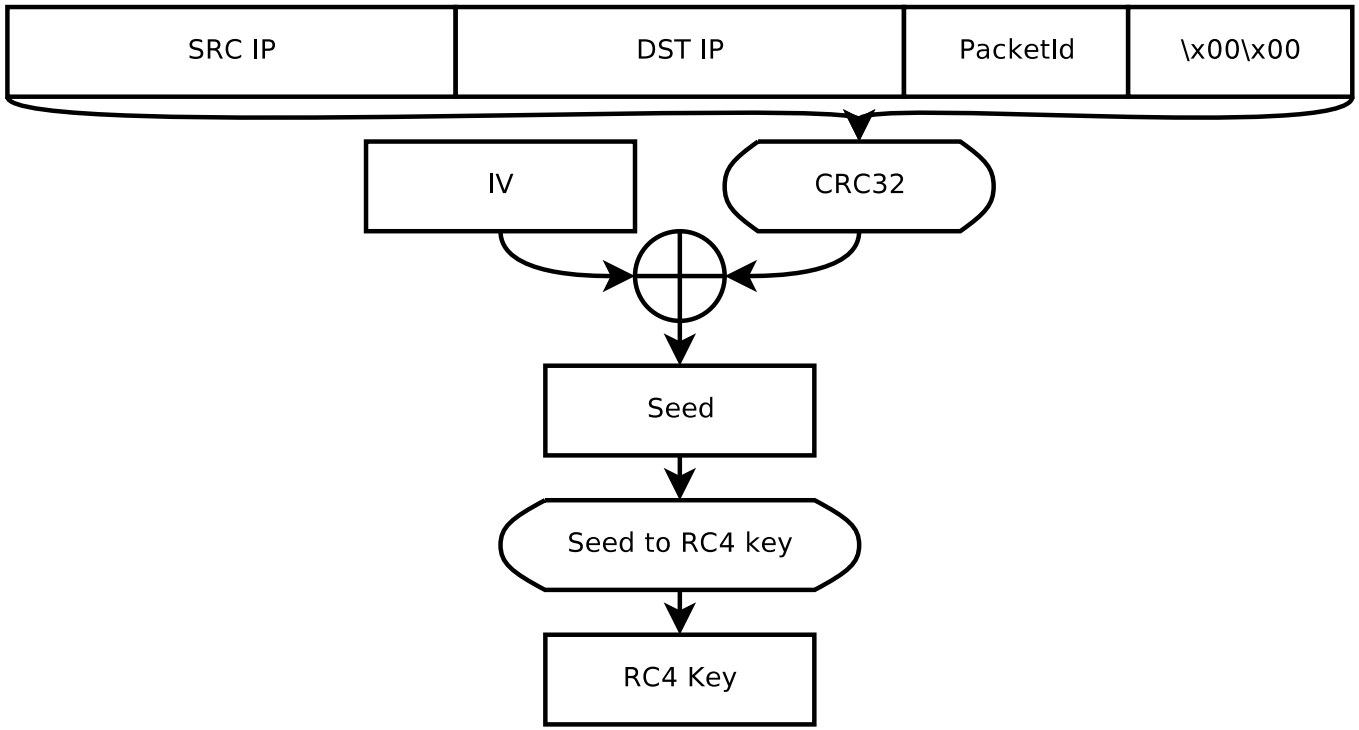
VoIP Packets

This encryption is different. Skype uses AES and only the sender/receiver can decrypt them. This is not a simple obfuscation.

Packet dissection



UDP packet deciphering



- The RC4 key is generated using src/dst IP plus packet ID.
- The clear payload is composed by objects containers, in which data are stored.
- Those data will be received by an object manager.

EADS CCR

Client authentication

Authority public key

13 trusted moduli (RSA). Size is between 1536 and 2048 bits.

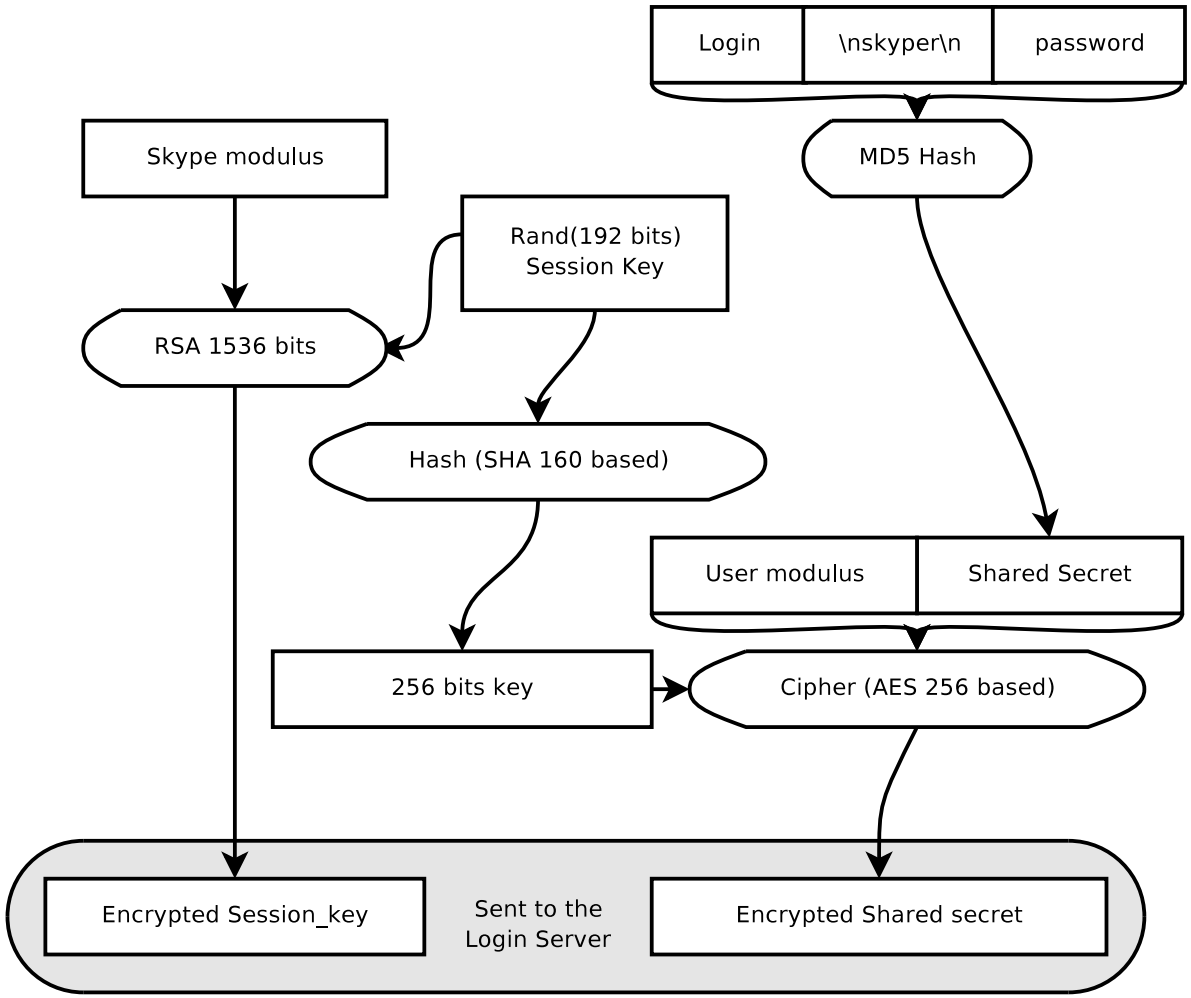
Client public key

- Each client generates its private/public key (RSA 1024 bits) at login time. It's a session RSA key
- A secret is shared between clients and the authority : the hashed password

Login mechanism

- The client generates a session key
- Encrypts the shared secret with it
- Then encrypts the session key with RSA (using a trusted modulus)
- If the authority passes the test, it signs the couple login/public key and sends it to Supernodes

Client authentication



EADS CCR

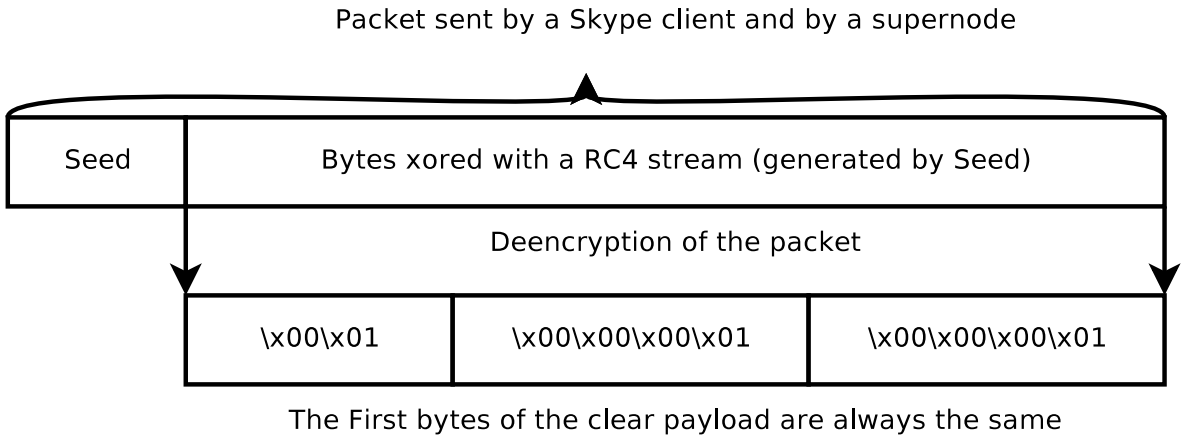
Skype detection

TCP Skype packet detection

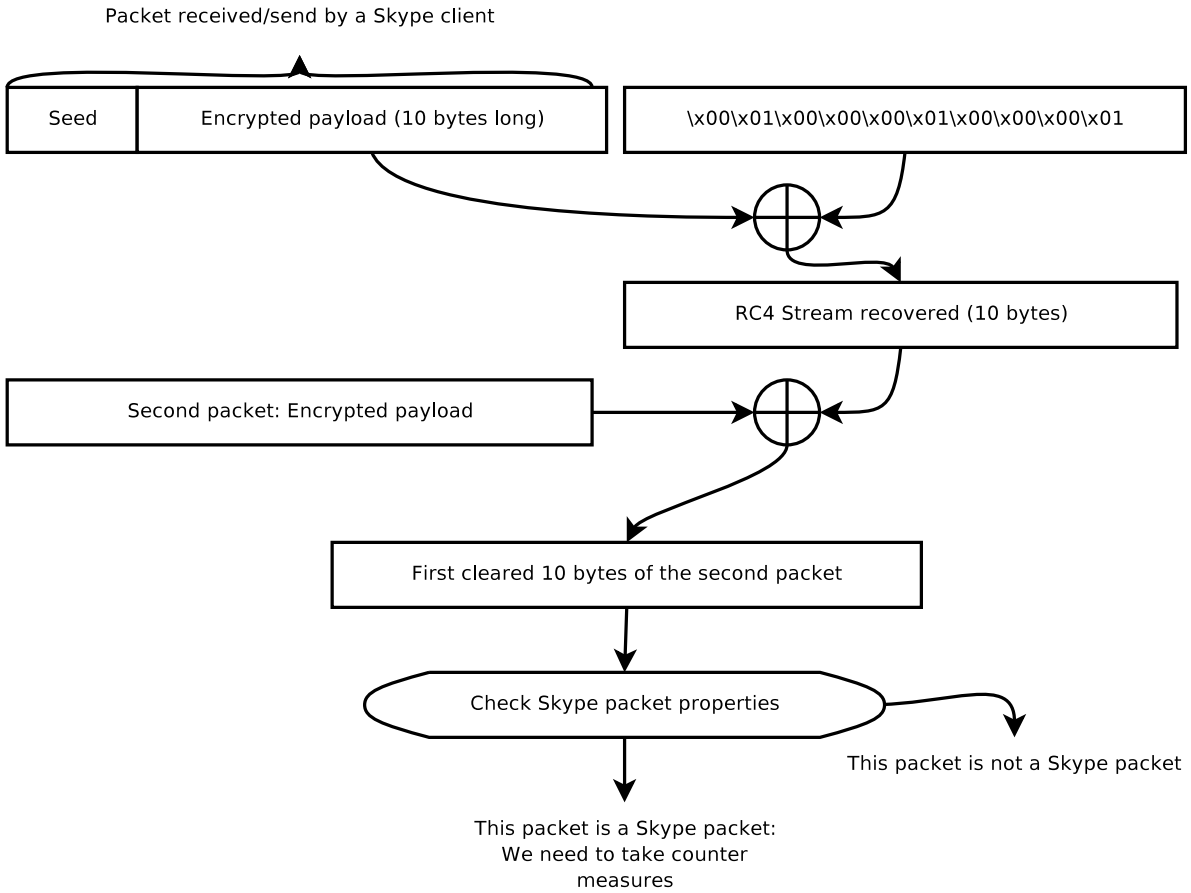
When a TCP session is established :

- Each machine sends its seed key to the other
- This seed will be used to generate a continues RC4 stream
- Except for the two first packets

$\Longrightarrow$This can be used to detect Skype connection by deciphering TCP packet without using internal decryption mechanism.



Skype TCP packet detection



Conclusion

Proprietary protocol

- Proprietary and obfuscated protocols don't prevent flaws
- It can only slow down the exploitation of it
- Worse, it may protect a 0-day

`http://seclists.org/lists/fulldisclosure/2005/Oct/0533.html`
`http://www.skype.com/security/skype-sb-2005-03.html`

Questions ?

EADS CCR