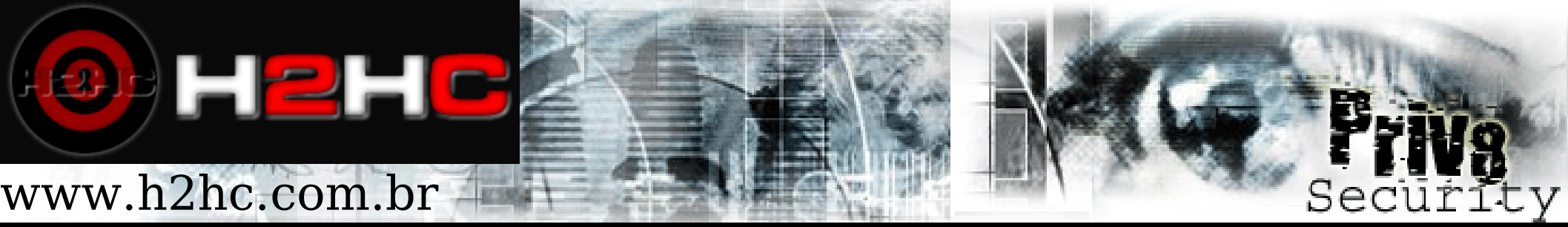
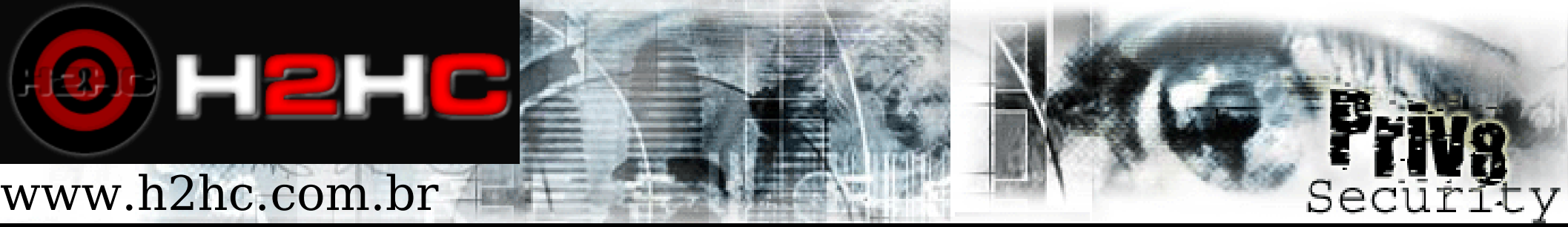


Polymorphic Attacks

Rodrigo Rubira Branco
rodrigo@firewalls.com.br
bsdaemon@bsdaemon.org



- Abstract: Some years ago, new idea has been inserted in the virus world: Code Polymorphism. This idea consists in codes with have conditions to mutate yourself, causing a new code all times it executes. Doing it, is impossible to detect virus signatures by simple pattern matching. In the hacking scene, this idea are now in the mind after RiX text to phrack. Understand how it works and how it can be done.

DIFFERS

"This demonstration doesnt have the intention to demonstrate how to construct IDS Signatures or how to evade it. It try to demonstrate how polymorphism techniques working, and show ideas to turn it a real thinking."

How it Works

---

call decryptor

---

shellcode

---

decryptor

---

jmp shellcode

How it Works

The decryptor are responsible to do the inverse process used to encode your shellcode.

This process can be, for example:

- ADD

- SUB

- XOR

- SHIFT

- Encryption (like DES)

When a call is made then the next address (in our case the shellcode's address is pushed on the stack. So, the decryptor can easily get the address of shellcode by poping a General Purpose Register. Then all the decryptor has to do is to manipulate the bytes of the shellcode and then jmp on the address of it. So the algo of the our encrypted shellcode should be like this:

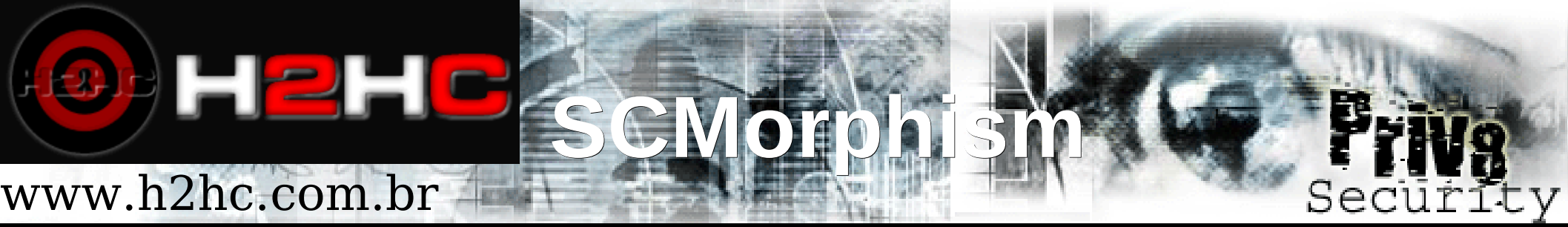
Ending..

You only need to concatenate the shellcode at the end of your decoder. Change the mov sizeof(shellcode), %cx bytes. Off-course you need to avoid any zero-bytes in the decryptor opcode because the system will understand it like a string terminator.

More improvements can be made, this is just a dirty code in order to show you how to implement, build a decryptor.

Difficulties

- Evit BADChars (like \r \t \0)
- Generate Alphanumeric Polymorphic Codes
- Eliminate constants in decoders
- Decoders build
- Mantain decoders for many platforms
- Insert “do nothing” instructions in the generated decoder/shellcode
- Optimize the decoder len



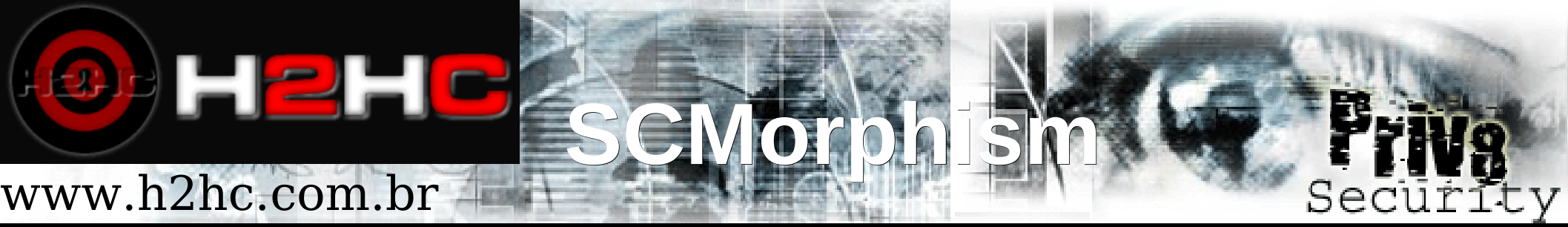
The intention is not only to encode the original shellcode with a random value and place a decoder in front or behind it. The tool place an randonly generated decoder in it.

SCMorphism can create XOR/ADD/SUB/ByteShift randomly decoders.

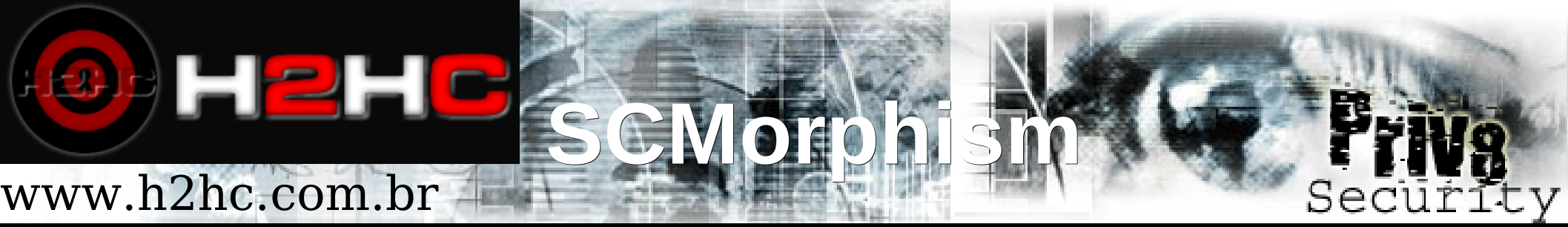
This decoders are splited in pieces (tks to Zillion@safemode.org for the idea used in your tool called s-poly).

Using this pieces, the tool can create randomly decoders, manipulatin this pieces and introducin "do nothing instructions".

This give to the user an modified encoded shellcode any time you use the tool.



Usage:
-f <shellcode file> <var name> <shellcode lenght>--> If it is .c ord . pl specify VAR Name and shellcode lenght
-t <type: 0 add, 1 sub, 2 xor, 3 byte shift>. If not specified, using random values
-n <number>. Value used in the xor, add or sub operation
-e --> Execute the generated shellcode
-s --> Stdout form: -s <file name> to a file, if not gived use console
-b <badchar1,badchar2,badcharn> --> Exclude this char from shellcode (\n \r and \0 are always excluded)
-c <archfile>,<osfile>,<typefile>
-o --> Optimize option. Be careful, it removes "do nothing" instructions and can be detected by IDS (the decoder are static in this case)-m --> Open Menu to choose shellcode to use. Used if you dont have the shellcode
-v --> Version and greetz information
-h



The do nothing instructions used and randomly generated decoders turn impossibble to write IDS signatures for this tool (or it give to admin a lot of false positives).

When you call the encode function it generate a random number to be used to choose the piece

The number choosed to encoding operations are used to perform operations in the shellcode (shift bytes, xor'd, increased or decremented).

Bad characteres can be choosed by the user (in the TODO you will see the future scmorphism will implement alphanumeric decoder generation).

If the user dont choose any badchar, the system will use \0,\r,\t (tks to Strider from priv8security).

The tool have included a "do nothing" array, with instructions dont offer any problems in the shellcode decoder process. This array (tks to Strider again) are used to generated randomly "do nothing" instructions to be used in the middle of decoder, and turn decoder generation possibilitys infinite (this is a real increment in the s-poly tool, with fixed number possibilitys and IDS detectable).

REFERENCES

http://cs.southwesternadventist.edu/
http://rosec.org
http://m00.void.ru
http://www.infosecwriters.com/text_resources/pdf/basics_of_sh
http://www.securityfocus.com/infocus/1768
http://www.priv8security.com
http://www.intel.com/design/Pentium4/documentation.htm#pa
http://www.phrack.org
http://www.bsdaemon.org
http://www.firewalls.com.br

Doubts?!?

Rodrigo Rubira Branco

rodrigo@firewalls.com.br

bsdaemon@bsdaemon.org