Hacking .NET Applications: The Black Arts

blackhat®
BRIEFINGS & TRAINING

USA + 2011
EMBEDDING SECURITY

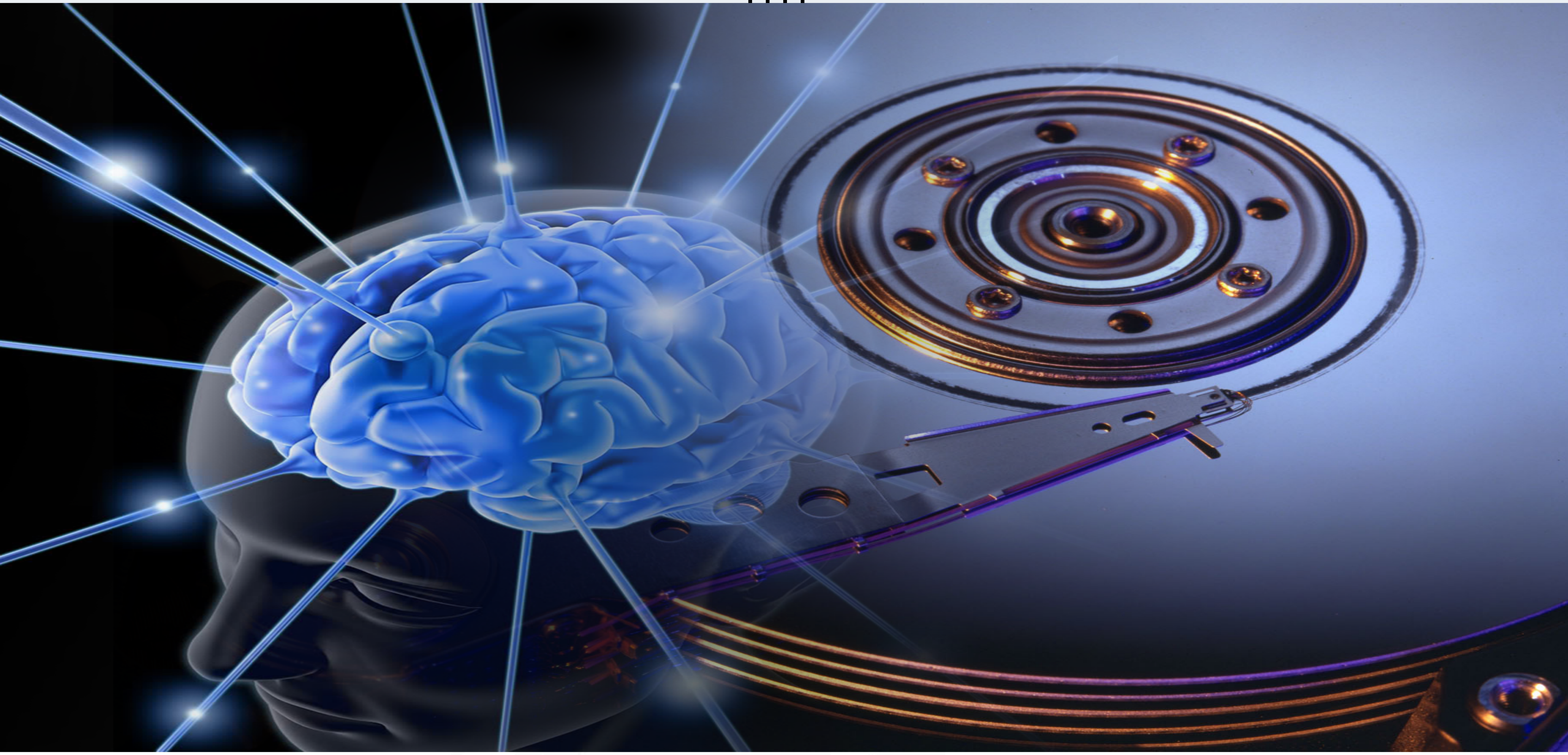
Jon McCoy
www.DigitalBodyGuard.com

THIS WILL COVER

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

Attacking/Cracking

IN MEM |||| ON DISK



blackhat USA + 2011

ATTACK OVERVIEW

Attack on Disk

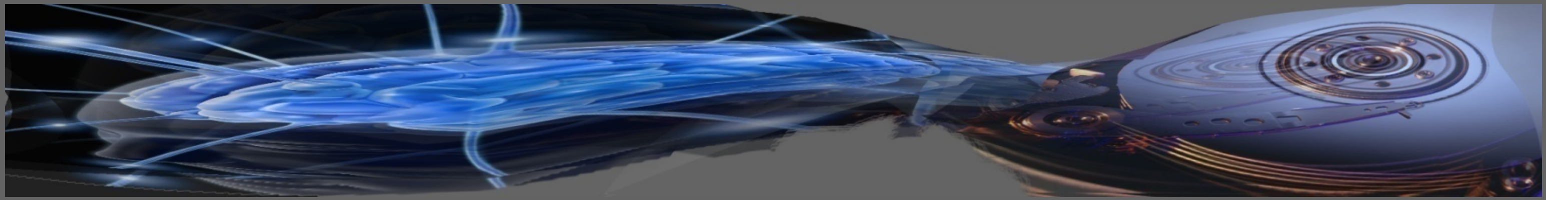
- Access Logic
- Infect Logic
- Hook Logic
- Decompile
- Recompile
- Debug

Attack in Memory/Runtime

- Inject Structure
- Navigate Structure
- Edit/Control Structure

Attack The Source

In Memory OR On Disk

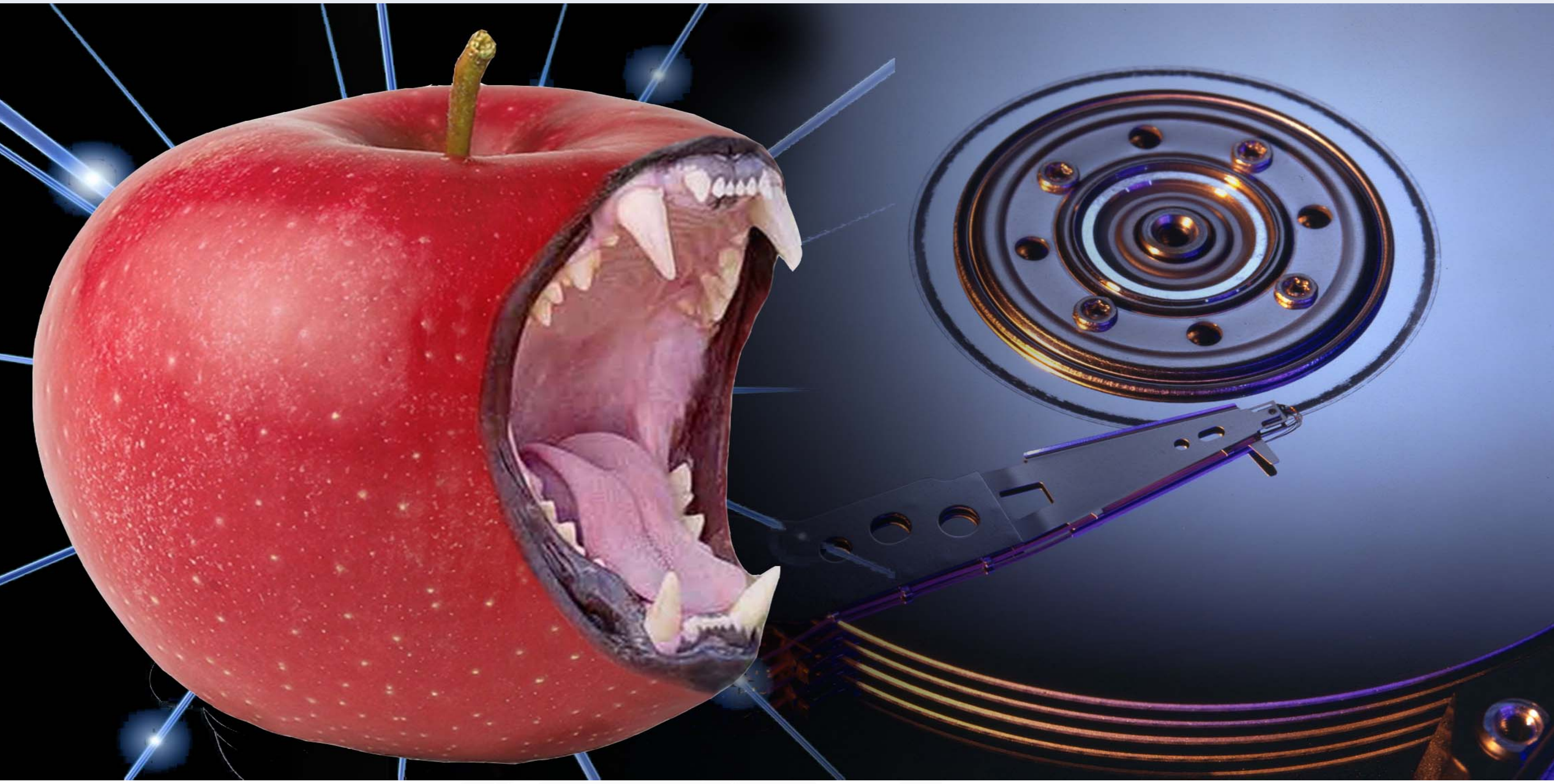


Find the weak spot

Subvert the Logic/State

Control what you need

ATTACKING ON DISK



101 - DECOMPILERS

DEMO

GrayWolf – IL_Spy – Reflector

101 - ATTACK ON DISK

- Connect/Open - Access Code
  - Decompile - Get code/tech
  - Infect - Change the target's code
  - Exploit - Take advantage
  - Remold Application - WIN

101 - Recon

EHSHELL

Windows Media Center

- .NET Framework
  Ver 3.5
- Un-0b**fu$**ca7ed
- Crash Reporting
  Watson
- Coded in C#

101 - Recon

- File Location

  C:\Windows\ehome\ehshell.dll

- StrongName KEY

  d:\w7rtm.public.x86fre\internal\strongnamekeys\fake\windows.snk

- Registry CurrentUser OR LocalMachine

  SOFTWARE\Microsoft\Windows\CurrentVersion\Media Center\

- Web Host Address

  www.microsoft.com/WindowsMedia/Services/2003/10/10/movie

CRACK THE APP

- Flip The Check
- Cut The Logic
- Return True
- Access Value
- Set Value is "True"

SET VALUE TO “TRUE”

~~bool Registered = false;~~

~~If(a != b)~~

CUT THE LOGIC

```
bool IsRegistered()
{
      Return “TRUE”
}
```

ACCESS VALUE

```
bool ValidPassword(int x)
{
    ShowKey(Pass);
    Return (x==Pass);
}
```

CRACK

CRACK the weak
Media Center

CRACK PASSWORD

```
public static bool CheckPin(string pin)
{
        ParentalControl.Settings.PIN = null;
        ParentalControl.Settings.Load();
        string text = ParentalControl.Settings.PIN;
        if (text == null)
        {
                return 1;
        }
        if (text.Length > 0)
        {
                if (text.get_Chars(0) == 58)
                {
                        goto Block_6;
                }
        }
        ParentalControlPin.StoreNewPin(text);
        return text == pin;
        Block_6:
        return text == ParentalControlPin.HashForPin(pin);
}
```

CRACK

DEMO

Attack the STRONG

"I'm sure they protected the registration check"

CRACK THE KEY

Complex Math	==	Complex Math
Public/Private	==	Change Key
Challenge	==	Make it EZ
3/B==Name*C	==	ASK what is /B?
Call Server	==	Hack the Call

COMPLEX MATH

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

CHANGE THE KEY

If you can beat them
Why join them

Key = "123456ABCDE"

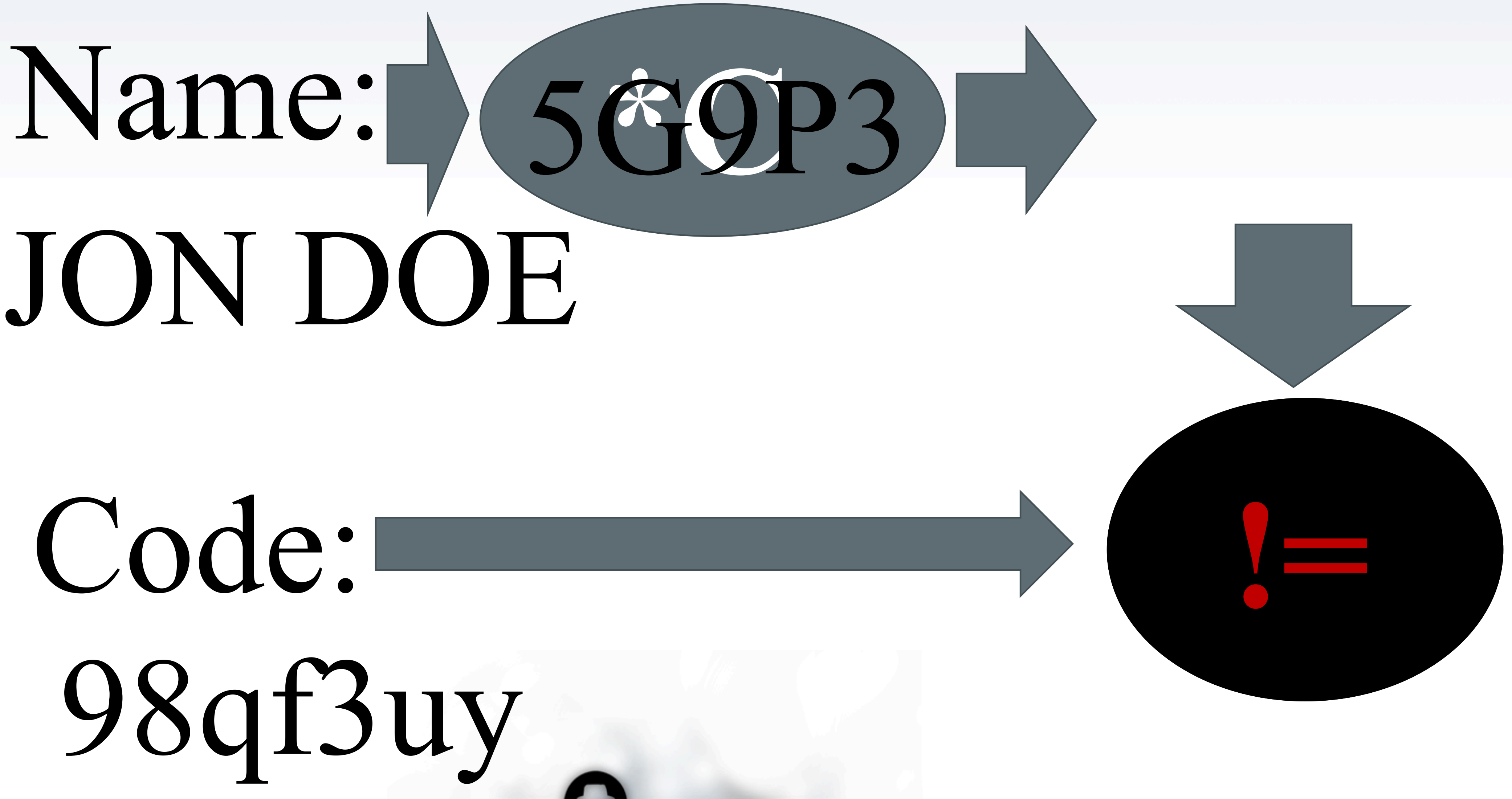
CHALLENGE

Complex Math

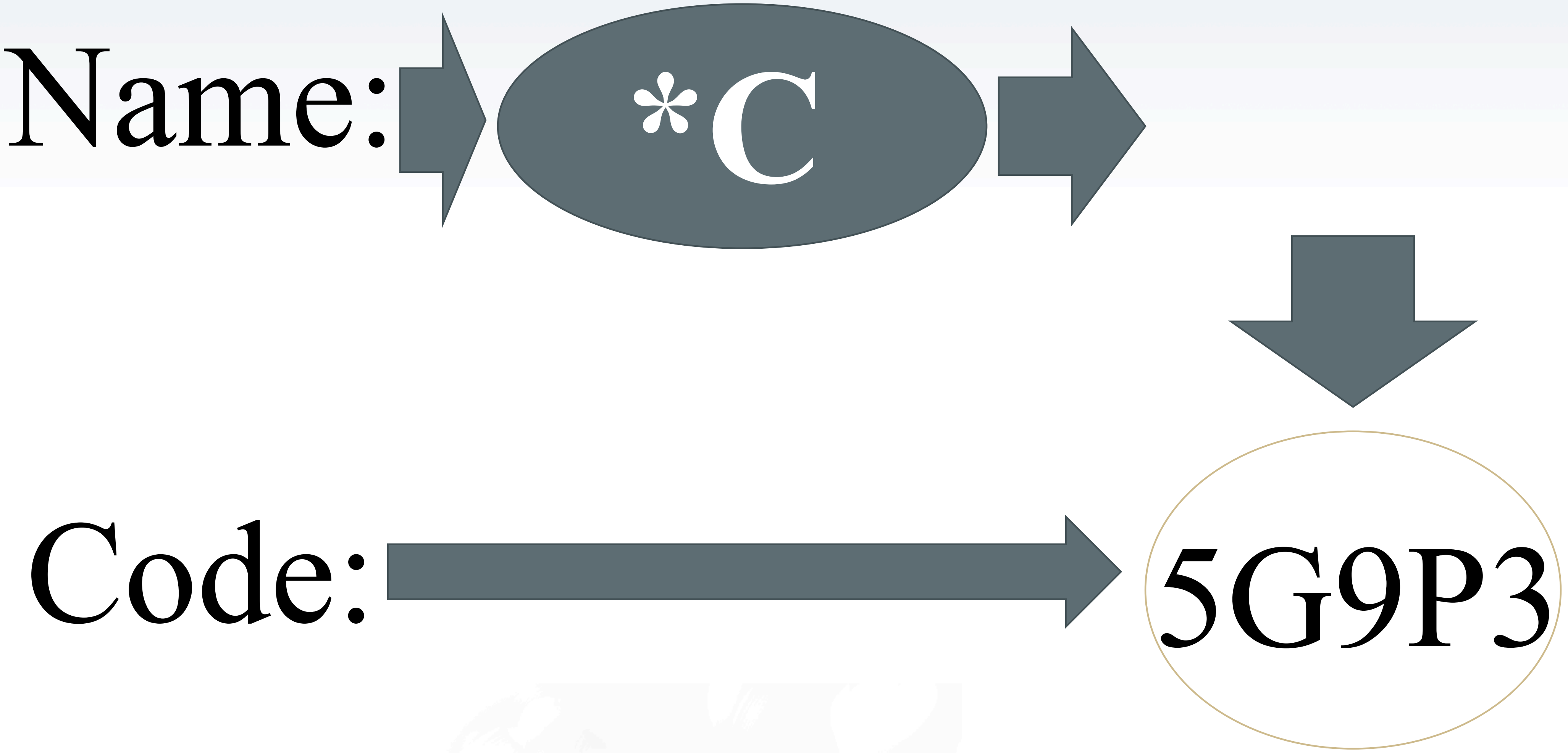
OR

Control the Challenge

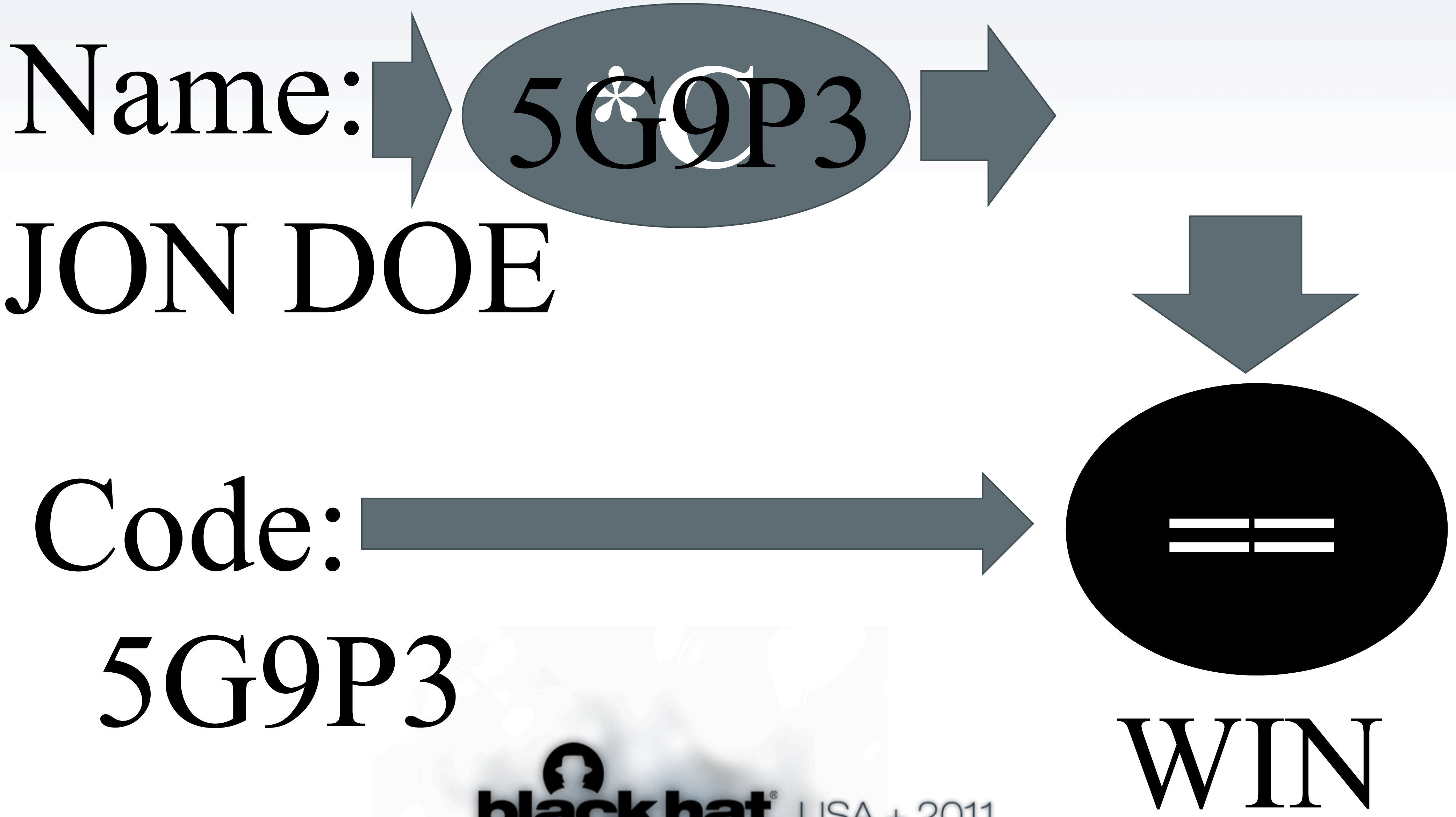
REG CODE



REG CODE



REG CODE



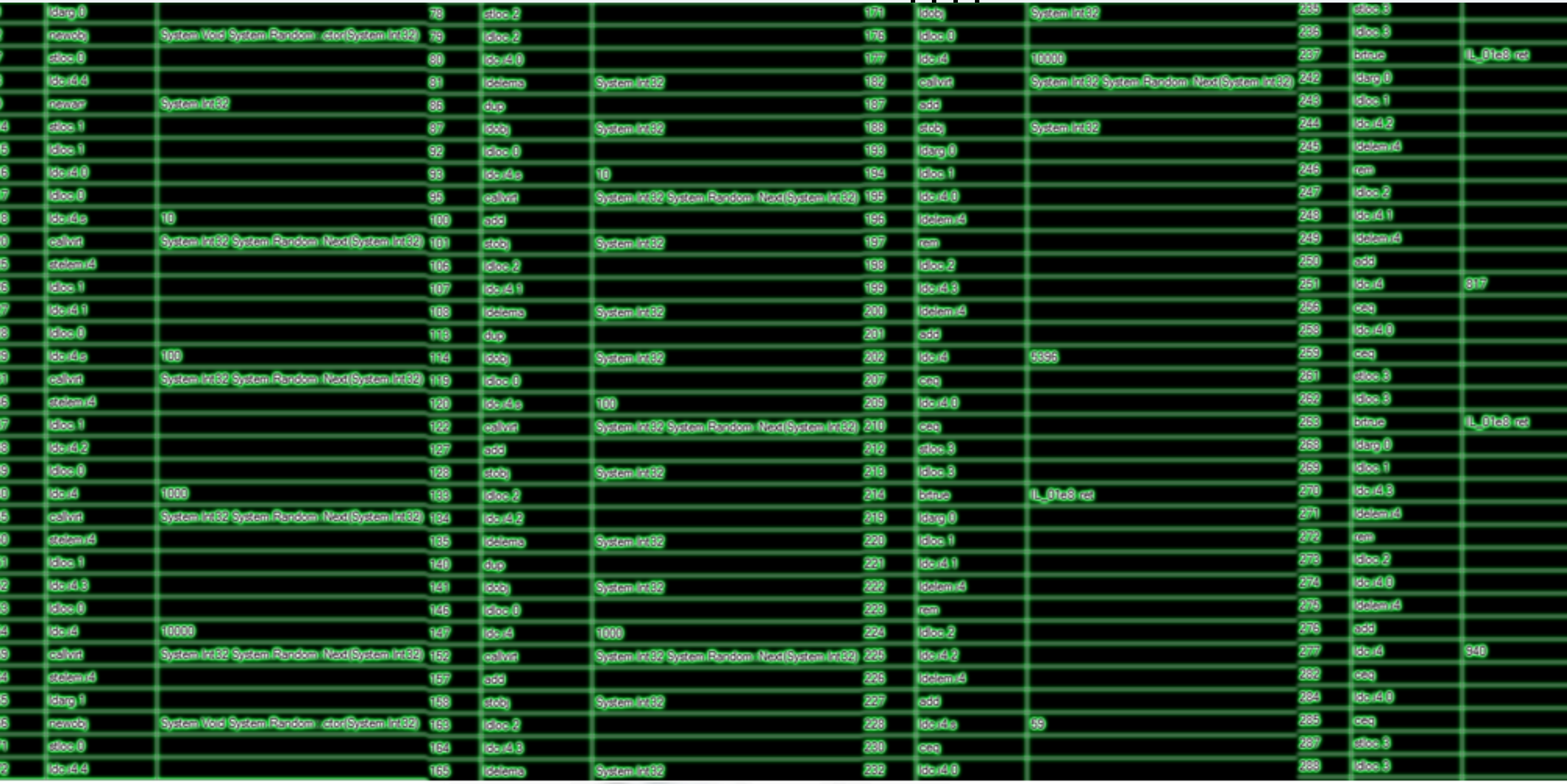
FAKE SERVER CALL

1. Seed the Request
2. Fake the Reply
3. Kill the Call

DEMO

CRACK A KEY

IL – Intermediate Language Code of the Matrix |||| NEW ASM



[illegible]	ldarg.0		78	stloc.2		171	ldobj	System.Int32	235	stloc.3	
[illegible]	newobj	System.Void System.Random::.ctor(System.Int32)	79	ldloc.2		176	ldloc.0		236	ldloc.3	
7	stloc.0		80	ldc.i4.0		177	ldc.i4	10000	237	brtrue	IL_01e8: ret
[illegible]	ldc.i4.4		81	ldelema	System.Int32	182	callvirt	System.Int32 System.Random::Next(System.Int32)	242	ldarg.0	
[illegible]	newarr	System.Int32	86	dup		187	add		243	ldloc.1	
4	stloc.1		87	ldobj	System.Int32	188	stobj	System.Int32	244	ldc.i4.2	
5	ldloc.1		92	ldloc.0		193	ldarg.0		245	ldelem.i4	
6	ldc.i4.0		93	ldc.i4.s	10	194	ldloc.1		246	rem	
7	ldloc.0		95	callvirt	System.Int32 System.Random::Next(System.Int32)	195	ldc.i4.0		247	ldloc.2	
8	ldc.i4.s	10	100	add		196	ldelem.i4		248	ldc.i4.1	
0	callvirt	System.Int32 System.Random::Next(System.Int32)	101	stobj	System.Int32	197	rem		249	ldelem.i4	
5	stelem.i4		106	ldloc.2		198	ldloc.2		250	add	
6	ldloc.1		107	ldc.i4.1		199	ldc.i4.3		251	ldc.i4	817
7	ldc.i4.1		108	ldelema	System.Int32	200	ldelem.i4		256	ceq	
8	ldloc.0		113	dup		201	add		258	ldc.i4.0	
9	ldc.i4.s	100	114	ldobj	System.Int32	202	ldc.i4	5396	259	ceq	
1	callvirt	System.Int32 System.Random::Next(System.Int32)	119	ldloc.0		207	ceq		261	stloc.3	
6	stelem.i4		120	ldc.i4.s	100	209	ldc.i4.0		262	ldloc.3	
7	ldloc.1		122	callvirt	System.Int32 System.Random::Next(System.Int32)	210	ceq		263	brtrue	IL_01e8: ret
8	ldc.i4.2		127	add		212	stloc.3		268	ldarg.0	
9	ldloc.0		128	stobj	System.Int32	213	ldloc.3		269	ldloc.1	
0	ldc.i4	1000	133	ldloc.2		214	brtrue	IL_01e8: ret	270	ldc.i4.3	
5	callvirt	System.Int32 System.Random::Next(System.Int32)	134	ldc.i4.2		219	ldarg.0		271	ldelem.i4	
0	stelem.i4		135	ldelema	System.Int32	220	ldloc.1		272	rem	
1	ldloc.1		140	dup		221	ldc.i4.1		273	ldloc.2	
2	ldc.i4.3		141	ldobj	System.Int32	222	ldelem.i4		274	ldc.i4.0	
3	ldloc.0		146	ldloc.0		223	rem		275	ldelem.i4	
4	ldc.i4	10000	147	ldc.i4	1000	224	ldloc.2		276	add	
9	callvirt	System.Int32 System.Random::Next(System.Int32)	152	callvirt	System.Int32 System.Random::Next(System.Int32)	225	ldc.i4.2		277	ldc.i4	940
4	stelem.i4		157	add		226	ldelem.i4		282	ceq	
5	ldarg.1		158	stobj	System.Int32	227	add		284	ldc.i4.0	
6	newobj	System.Void System.Random::.ctor(System.Int32)	163	ldloc.2		228	ldc.i4.s	59	285	ceq	
1	stloc.0		164	ldc.i4.3		230	ceq		287	stloc.3	
2	ldc.i4.4		165	ldelema	System.Int32	232	ldc.i4.0		288	ldloc.3	

IT CAN’T BE THAT EZ

NO

PROTECTION ON DISK

- Protection - Security by 0b$cur17y
  - Code Obfuscation
  - Logic Obfuscation
  - Shells / Packers / Encrypted(code)
  - Unmanaged calls…………..

SHUTDOWN

Decompilation

PROTECTION ON DISK
0b**fu**$ca7ed

```
public static bool ⊠⊠⊠()
{
        try
        {
                bool flag = ( & 4) == 4;
        }
        catch (Exception exception)
        {
                ⊠⊠⊠.⊠⊠(arg_0F_0, box(Application));
                throw;
        }
        return flag;
}
```

PROTECTION ON DISK

- Protection – Security by security
  - Signed code (1028 bit CRYPTO)
  - Verify the creator
  - ACLs………. M$ stuff

This can SHUTDOWN Tampering

UNPROTECTED/PROTECTED

IT CAN~~'T~~ BE THAT EZ

~~NO~~ YES

SIGNED CODE HACKING

Signed code is based on

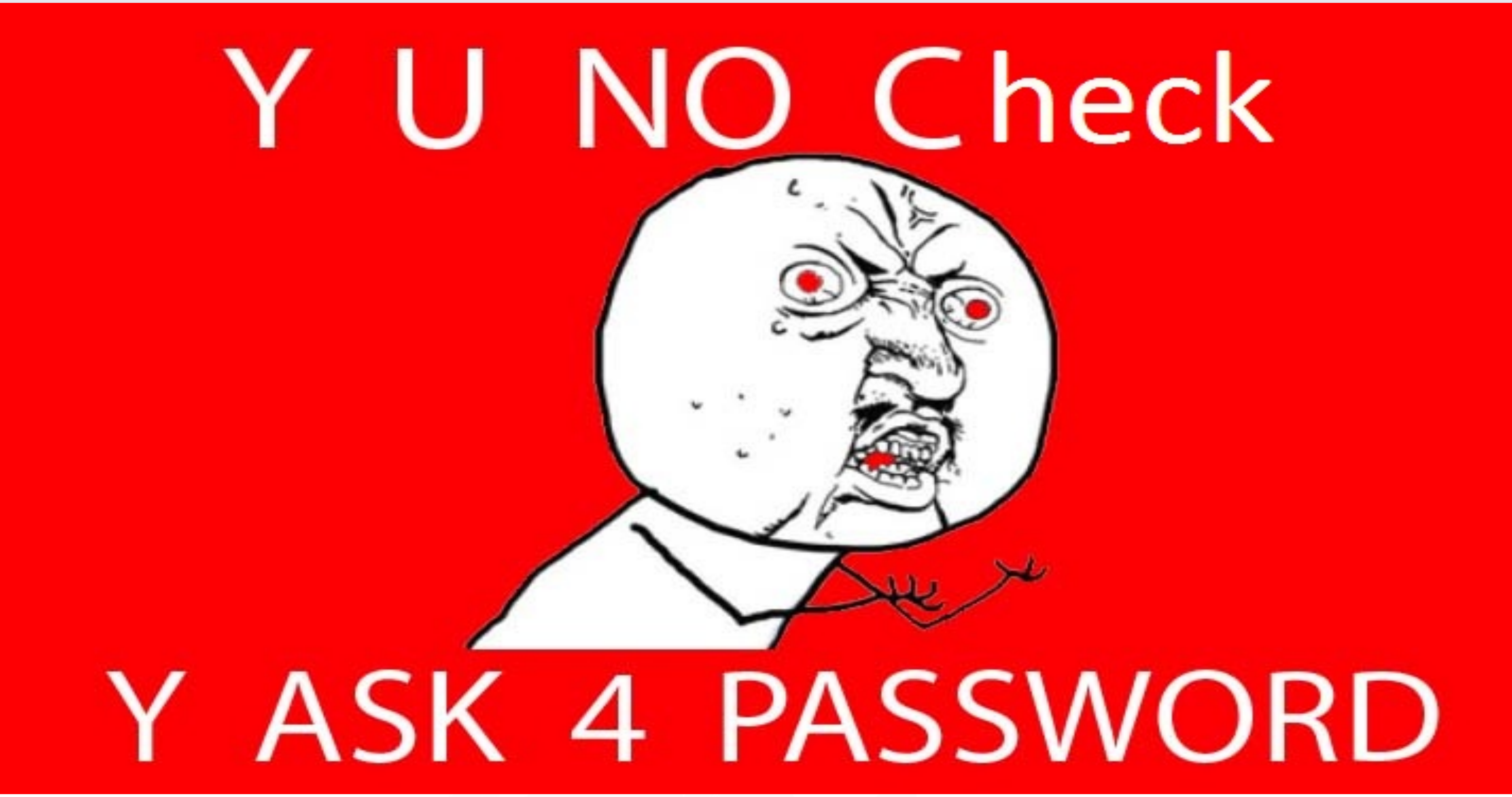
- Private Key - 1028 bit
- Signed Hash of Code
- ………..

SIGNED CODE CHECKING IS OFF BY DEFAULT

FAKE SIGNED DLL

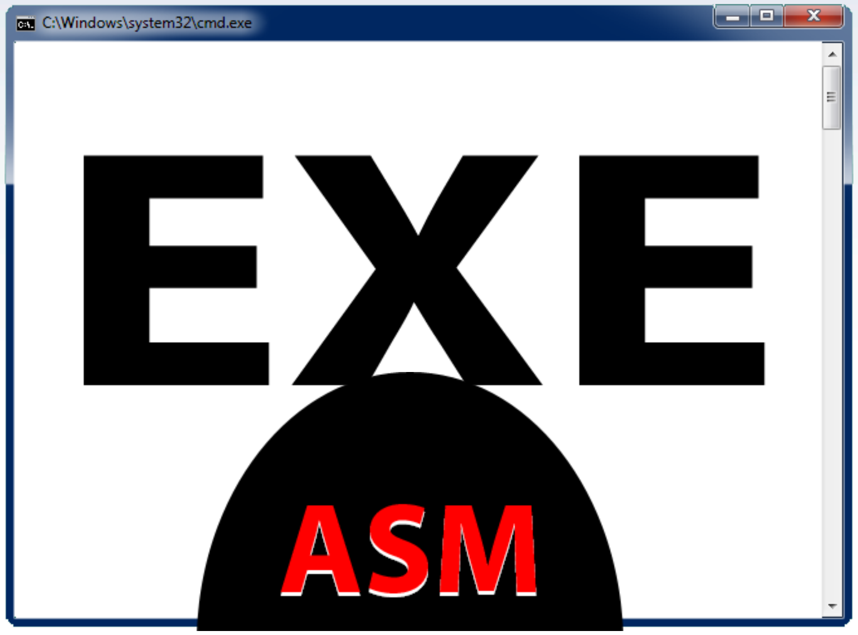
genuine Microsoft software
C:\Windows\system32\cmd.exe
EXE
PRIVATE
DLL
public

FAKE SIGN DLL/EXE



NEW ATTACK VECTOR

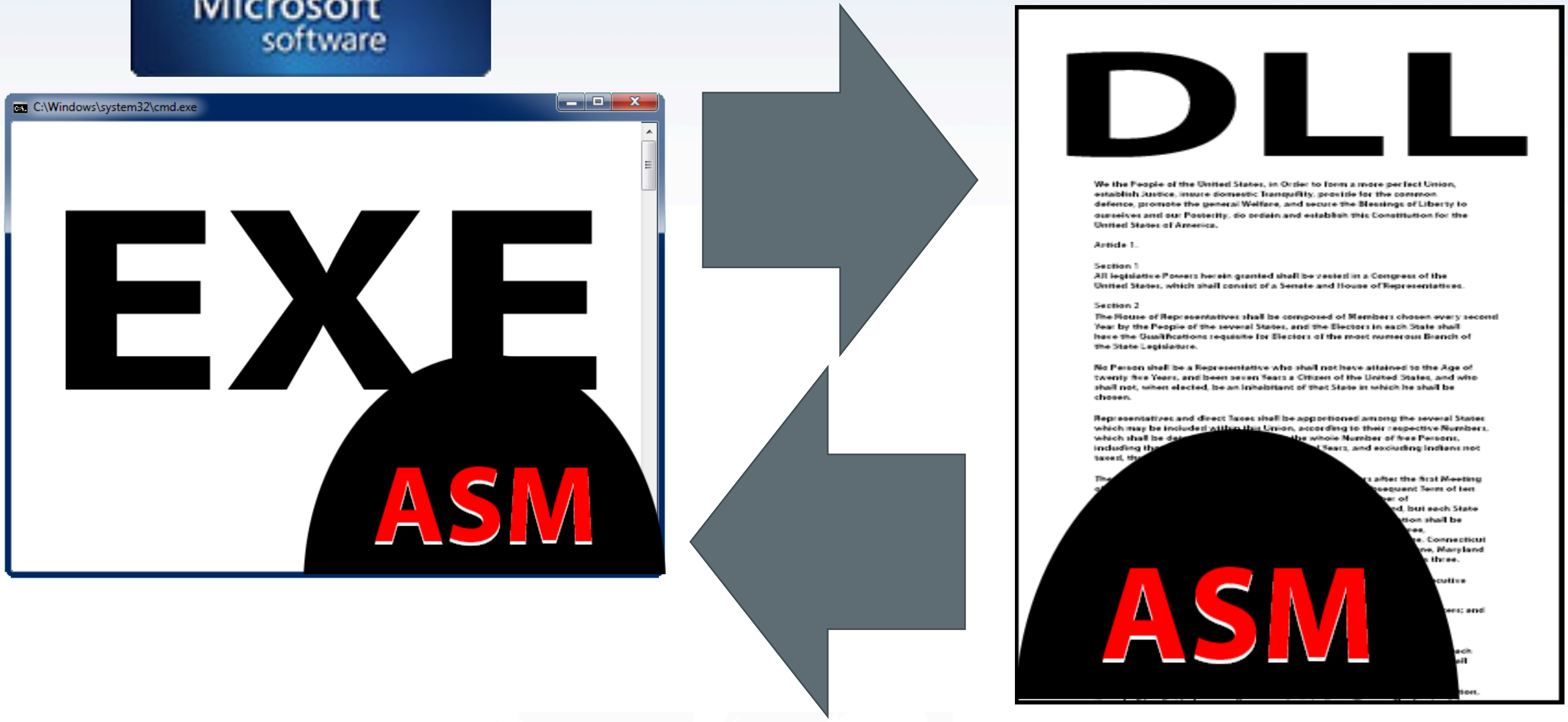
ASM THE OLD IS NEW



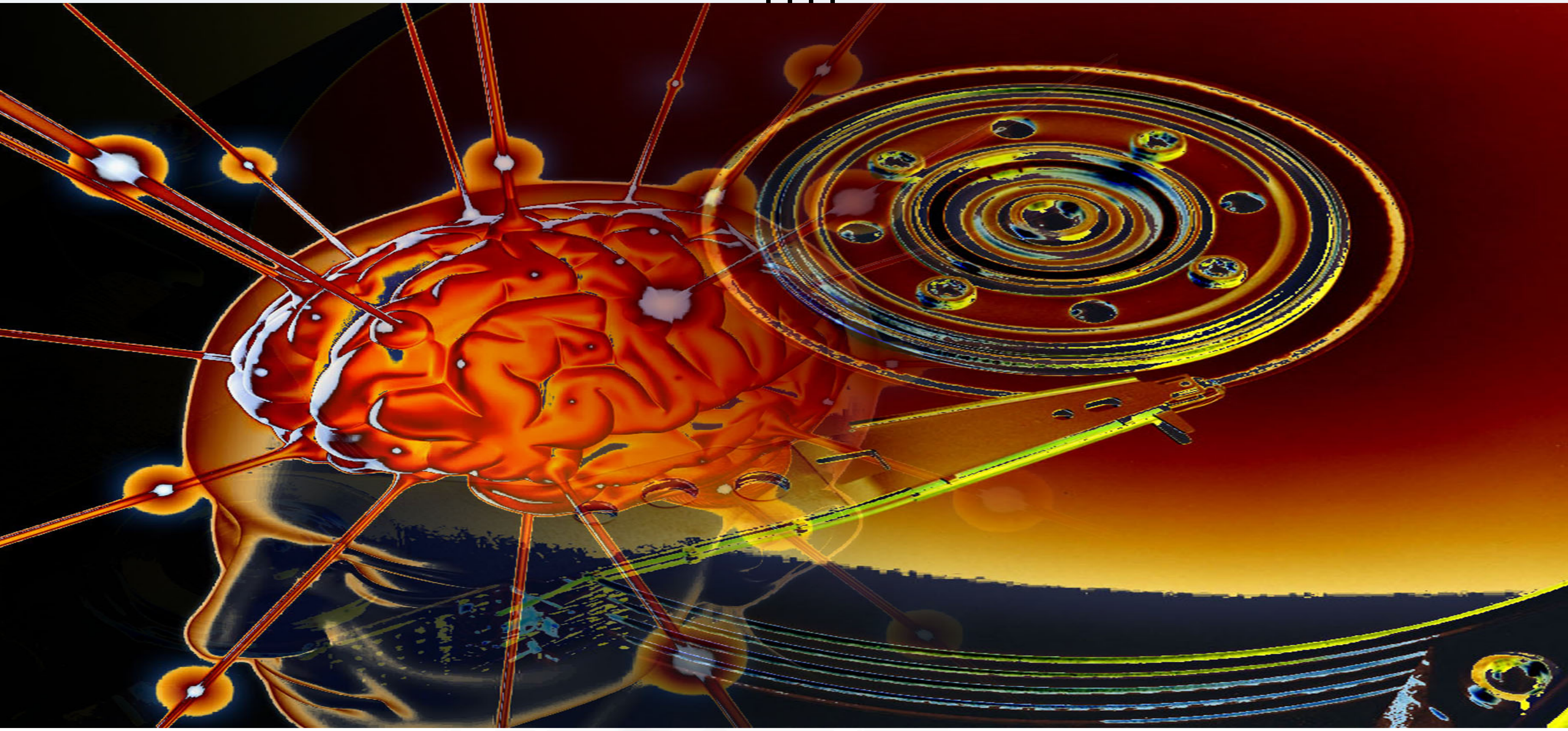
- Shell Code - ASM
- UNmanaged
- NO .NET Security
- ....

Attack from ASM(unmannaged)
The Gate is Down

FAKE SIGNED DLL

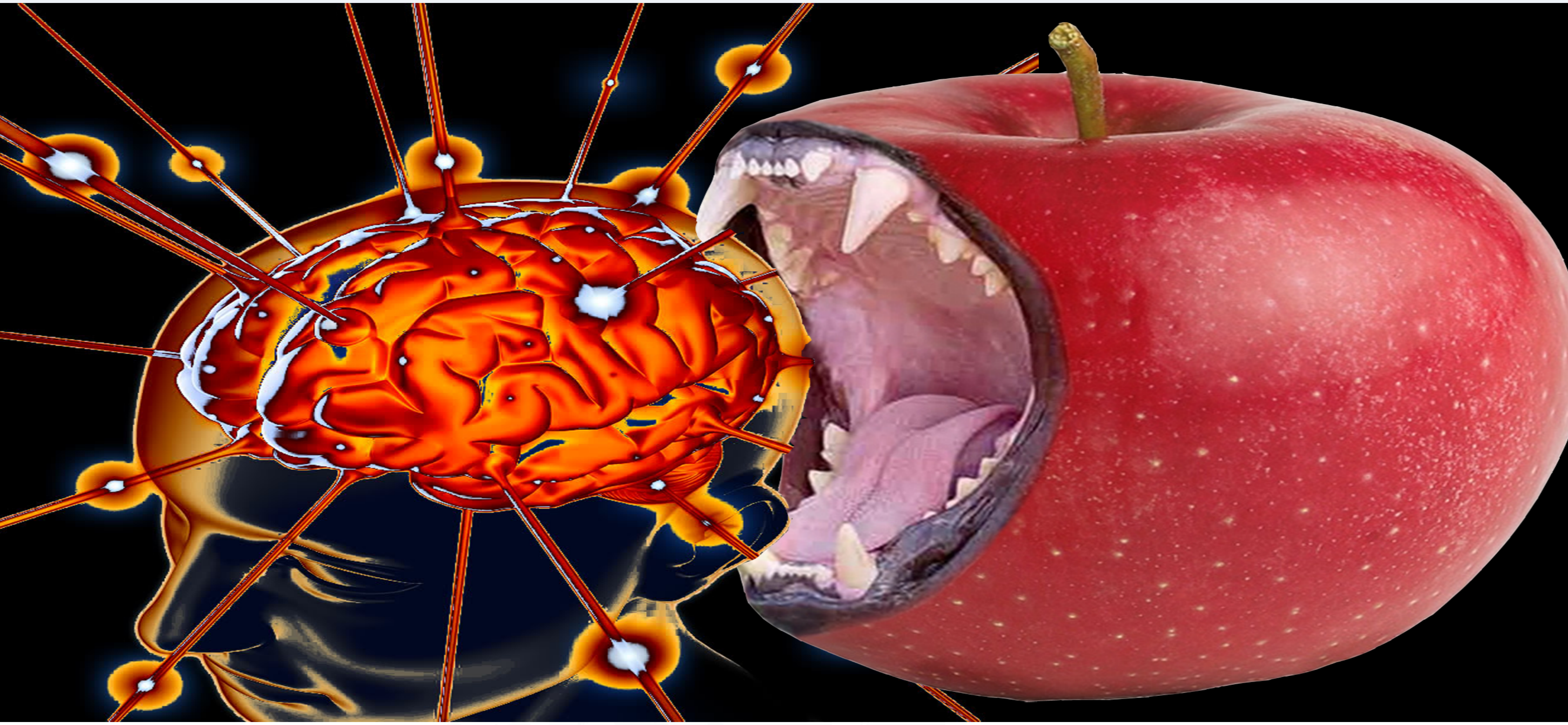


Attacking / Cracking
In MEM |||| On Disk



blackhat USA + 2011

ATTACKING .NET APPLICATIONS: AT RUNTIME



blackhat USA + 2011

ATTACKING APPS

- Gain Full Access
  - Reverse Engineer
    - Attack (on Disk or in MEM)
      - Take out the “Security”
        - Control the Program

DEMO: GOD MODE

Inject and Control

PAST TALKS

Hacking .NET Application:
A Runtime Attack

Control the Runtime
Control the Application

blackhat USA + 2011

IF YOU'RE NOT A HACKER WHY SHOULD YOU CARE?

Defend your Applications

Defend your Systems

Verify your Tools\Programs

VERIFY YOUR APPLICATIONS

What is the Crypto & KEY

What info is it sending home

Does it have Backdoors?

Is your data Secure?

REVERSE ENGINEERING

What is going on?

What technology is used?

Any MaLWare?

AM I safe?

REVERSE ENGINEERING

Hack your applications

Don't be helpless

DON'T LOOK

blackhat USA + 2011

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 / Serial Number
- Security / System Data

blackhat USA + 2011

KEY

The Crypto Key is

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

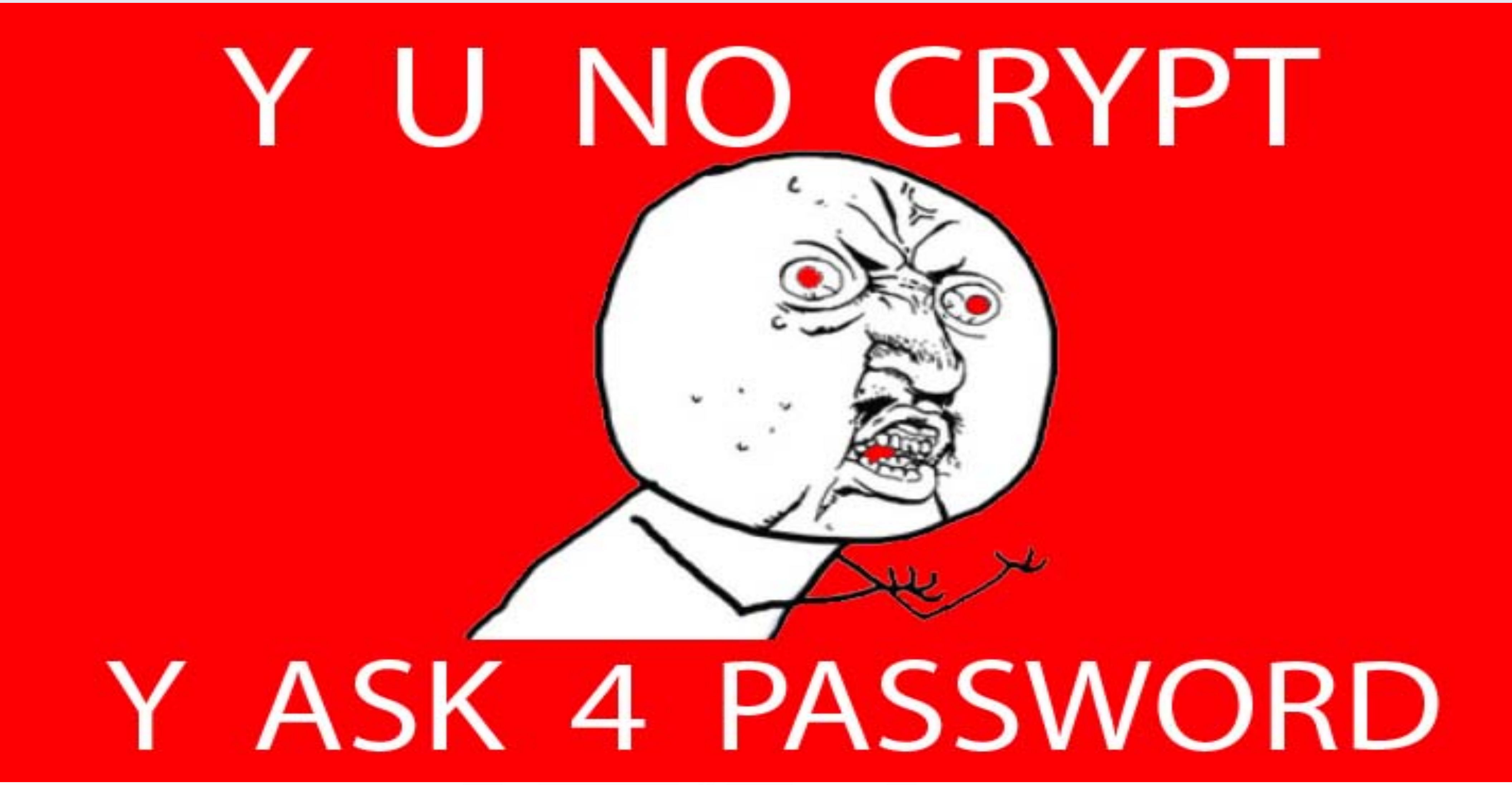
CRYPTO

The Crypto is

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

blackhat USA + 2011

BAD SOFTWARE



THE OLD IS NEW AGAIN ASM-SHELLS

- Pointers in .NET
  - What are they good for?
  - Are they safe?
  - What about the Runtime?
  - So ASM-Shells….

MALWARE T1M3

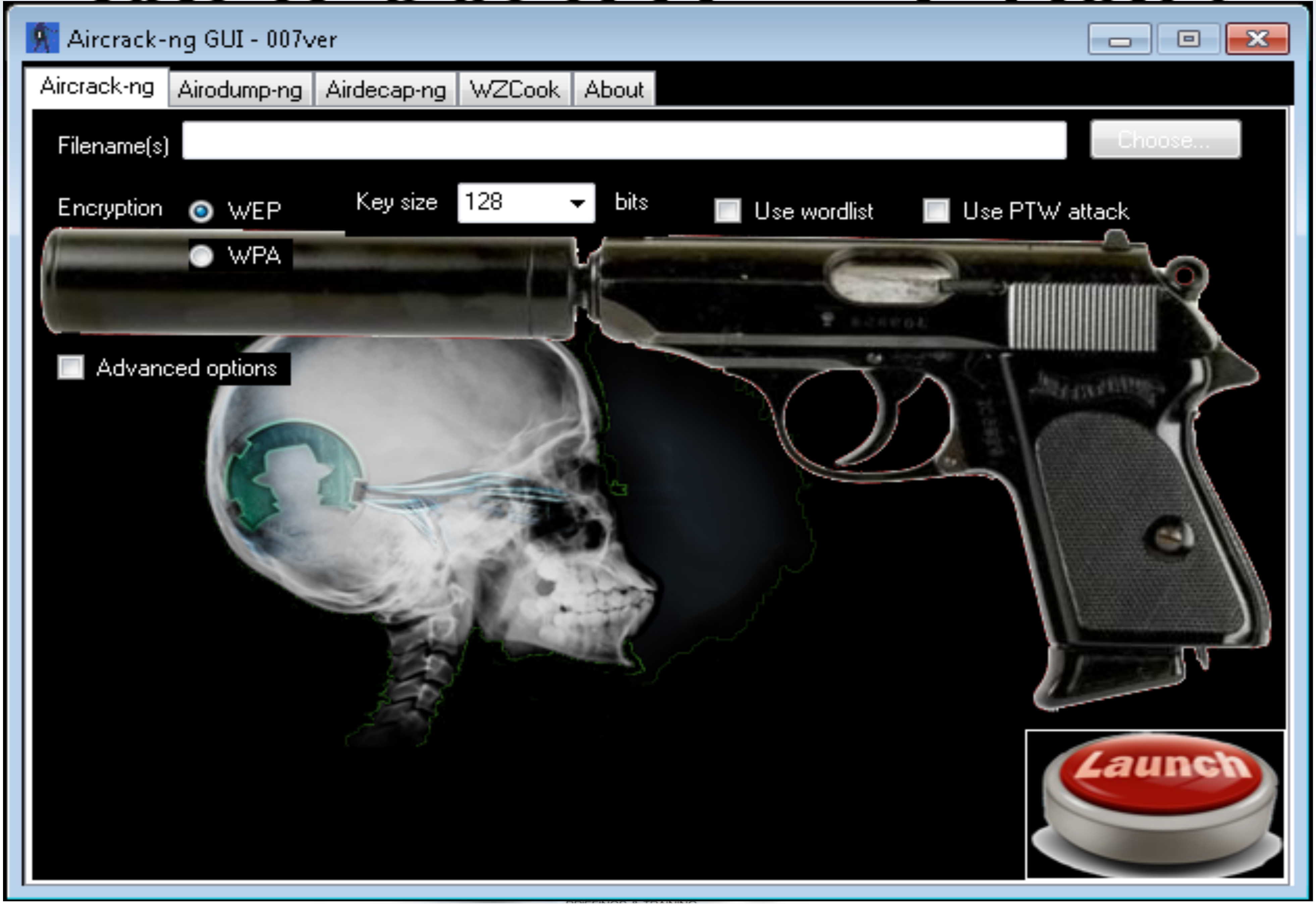
Protection (Shell Crypto)

Attack (Unmanaged Calls)

Protection (Obfuscated Code)

Fake (Signed DLL Protection)

MALWARE T1M3



Aircrack-ng GUI - 007ver

Filename(s)

Encryption [✓ WEP] WPA

Key size 128 bits

Use wordlist

Use PTW attack

Advanced options

MALWARE HIDE

REUS RGET

- Intelligent names
- Code style
- Don’t use loops
- Don’t use one area for your Vars
- Access the normal program

- Link to Events
- Use Timers
- Call back into your target
- Spread out your Vars and code

MALWARE FIGHT

MALWARE FIGHT

- TD3Ms
- fDP3St
- VfSdaI

Protect Me! 2010

- Rijndael
- Salt
- Good VI

0b$cur17y

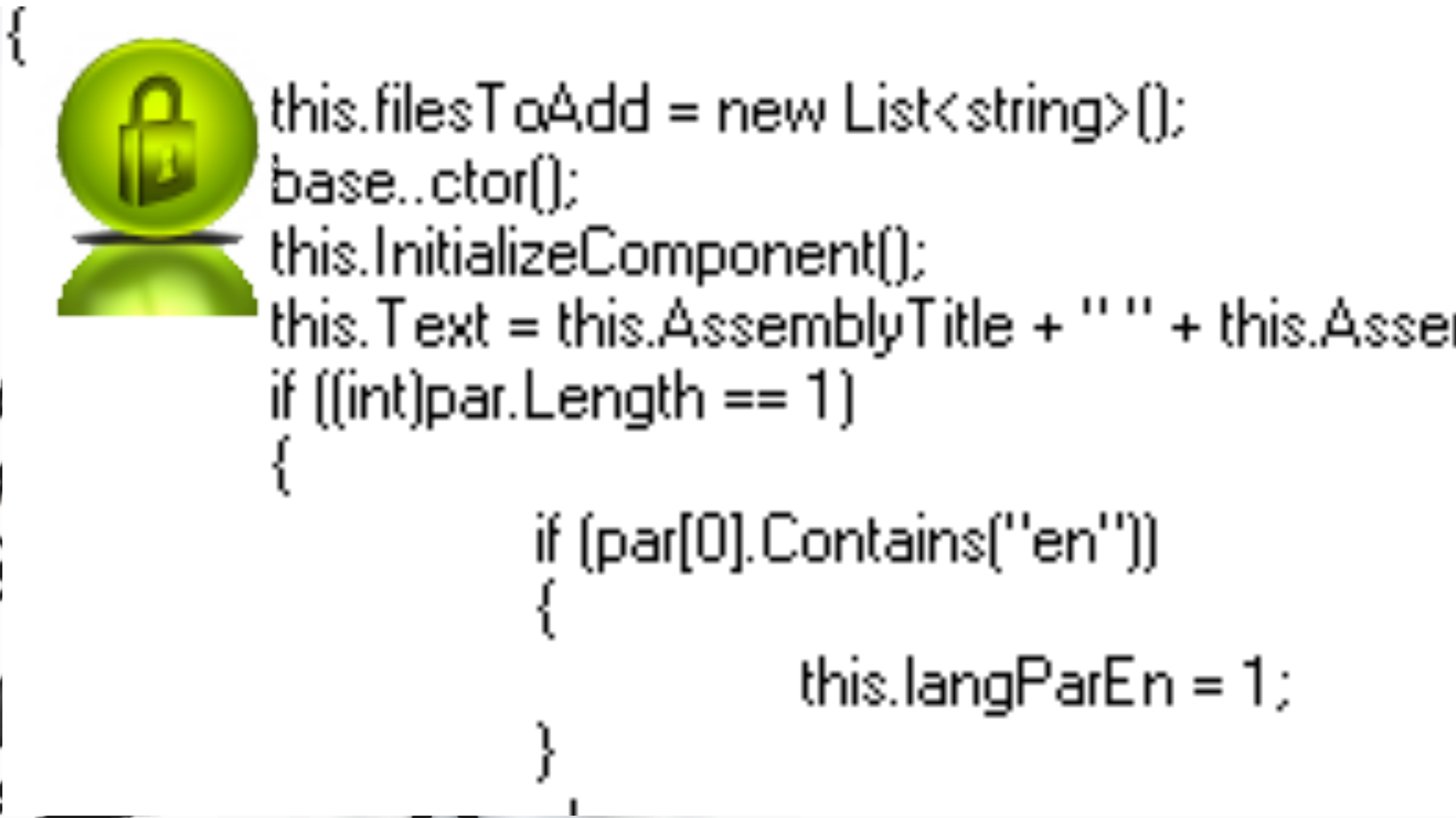
Androsa FileProtector

MALWARE FIGHT

MALWARE FIGHT

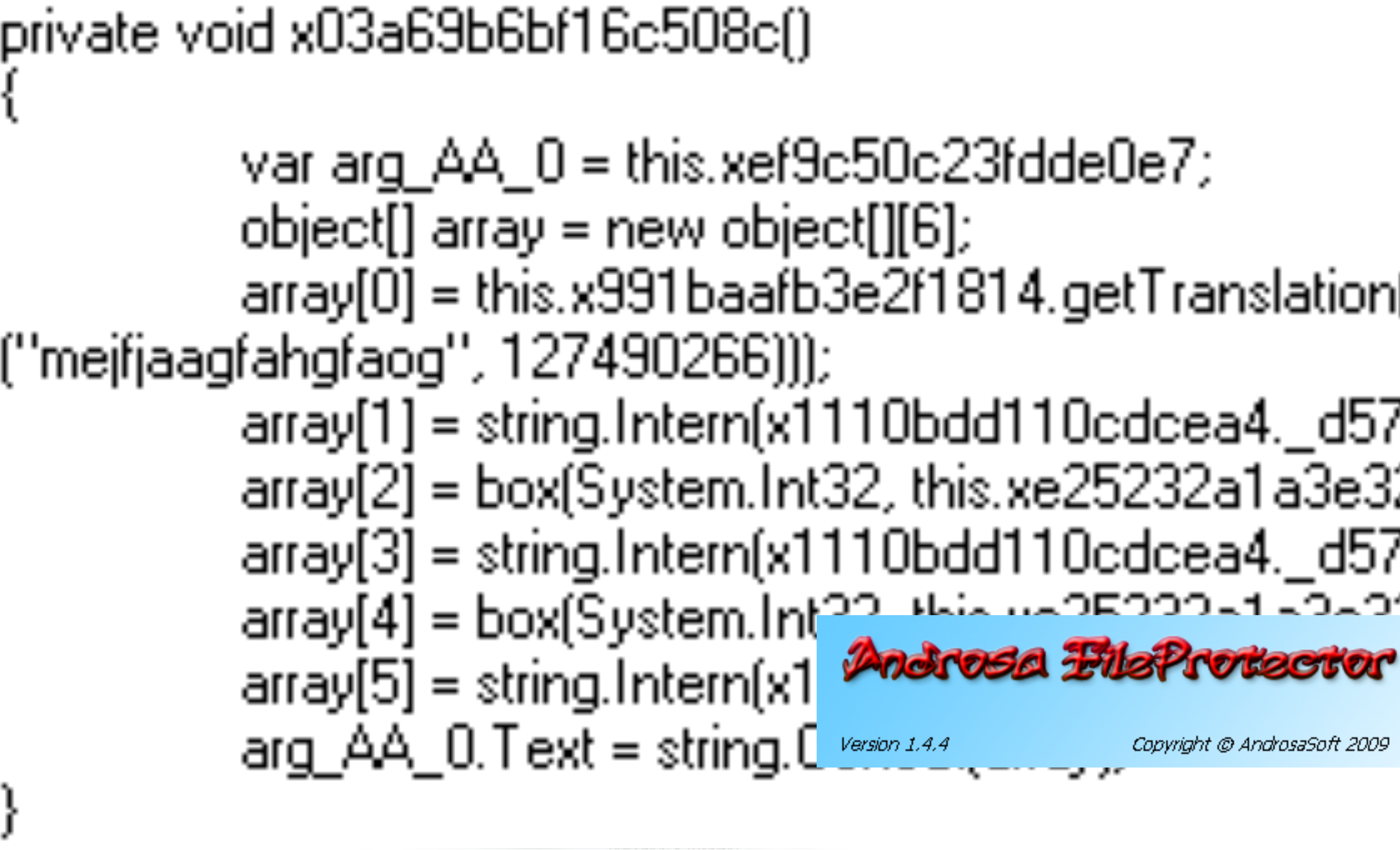
MALWARE FIGHT
W

Protect Me! 2010



```
{
    this.filesToAdd = new List<string>();
    base..ctor();
    this.InitializeComponent();
    this.Text = this.AssemblyTitle + " " + this.Asse
    if ((int)par.Length == 1)
    {
            if (par[0].Contains("en"))
            {
                    this.langParEn = 1;
            }
```

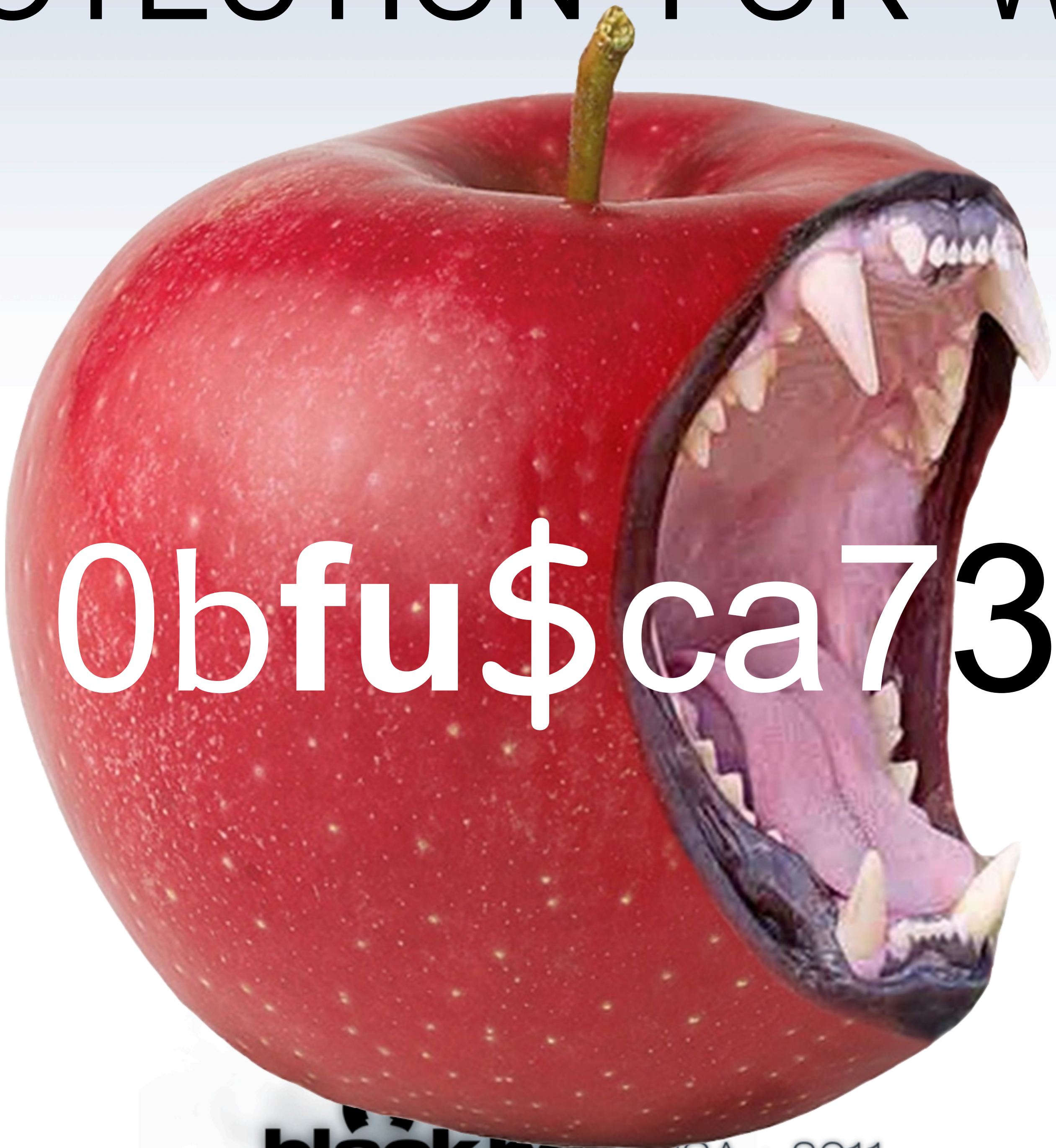
Androsa FileProtector



```
private void x03a69b6bf16c508c()
{
		var arg_AA_0 = this.xef9c50c23fdde0e7;
		object[] array = new object[][6];
		array[0] = this.x991baafb3e2f1814.getTranslation(
("mejfjaagfahgfaog", 127490266)));
		array[1] = string.Intern(x1110bdd110cdcea4._d57
		array[2] = box(System.Int32, this.xe25232a1a3e3
		array[3] = string.Intern(x1110bdd110cdcea4._d57
		array[4] = box(System.Int
		array[5] = string.Intern(x1
		arg_AA_0.Text = string.C
}
```

MALWARE FIGHT

PROTECTION FOR WHO?



WHAT M$ DID **RIGHT**

Un-obfuscated Code
- ∑ Good for user security
  - ∑ User can see what they are running

.NET Framework Security
- ∑ Targeted Security Access
  - ∑ Protect the Computer from the app

Giving Reduced Rights Inside Code
- ∑ Put venerable code in a box
  - ∑ Mitigate Risk, Segment Risk

WHAT M$ DID WRONG

MixModeCode – Bad for security
- This allows unmanaged code
  - This breaks out of .NET security

GAC & Native Image Override
- Removes ability to secure code

Not Hash Checking Code
- Good for hackers

ATTACKING APPS

- Read my papers: Reflections Hidden Power & Attacking .NET at Runtime
- Watch 2010 Presentations on Attacking .NET DefCon 18, AppSec-DC, DojoCon
- Look up Presentations and Research from Andrew Willson, Erez Ezule, Arndet Mandent
- Use tools: Visual Studio/MonoDev Reflector/ GrayWolf/ILspy/…/ILASM/ILDASM

FIN

MORE INFORMATION @:

www.DigitalBodyGuard.com

FIN = 1

HACKER VS ATTACKER