BACKGROUND

- Why .NET
- Countermeasures
- Skill Level Needed
- Will this work on every .NET application
THIS WILL COVER

- How-To Attack .NET Applications
- Tools and Methodology of Attacking
- Overcome “secure” .NET Applications
- Building KeyGen/Crack/Hacks/Malware
- Reverse Engineering for Protection
Attacking/Cracking
IN MEM ||| ON DISK
ATTACK OVERVIEW

Attack on Disk
- Access Logic
- Infect Logic
- Hook Logic

Attack in Memory/Runtime
- Inject Target App
- Navigate Structure
- Edit/Control Structure
- Decompile
- Recompile
- Debug
**Attack The Source**

<table>
<thead>
<tr>
<th>In Memory</th>
<th>OR</th>
<th>On Disk</th>
</tr>
</thead>
</table>

Do your Reconnaissance
Find the weak spot
Subvert the Logic/State
Control what you need
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
THE WEAK SPOTS

- Flip The Check
- Set Value is “True”
- Cut The Logic
- Return True
- Access Value
If (a != b)

If (a == b)

bool Registered = false;
bool IsRegistered()
{
    Return "TRUE";
}

bool IsValidKey(string x) {
    Return "TRUE";
}

string sqlClean(string x) {
    Return x;
}
bool ValidPassword(int x) {
    ShowKey(Pass);
    Return (x==Pass);
}
ATTACK SECURITY

Microsoft

Media Center
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);
}
public static bool CheckPin(string pin)
{
    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);
REGISTRATION CHECK

- KeyGens
- Cracks
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
   SystemID = 123456789
2. Fake the Request
   Reg Code = f3V541
3. Fake the Reply
4. Win

*Registered = True*
REG CODE REPLAY

Name: JON DOE

Code: 98qf3uy

5G9P3

FAIL
Name: *C
Code: 5G9P3
Name: JON DOE
Code: 5G9P3
REG CODE REPLAY

WIN
COMPLEX MATH

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

CRACK A KEY
**IL – Intermediate Language**

**Code of the Matrix | NEW ASM**
IT CAN’T BE THAT EZ

NO
PROTECTION ON DISK

Protection - Security by $\text{Ob$cur17y}$
- Code Obfuscation
- Logic Obfuscation
- Shells / Packers / Encrypted(code)
- Unmanaged calls

Try to SHUTDOWN
Decompilation
PROTECTION ON DISK

```csharp
public static bool protectedOnDisk()
{
    try
    {
        bool flag = ( & 4 ) == 4;
    }
    catch (Exception exception)
    {
        Exception exception;
        throw;
    }
    return flag;
}
```
PROTECTION ON DISK

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

Try to SHUTDOWN
Tampering
STRONG NAME

Simple Name
Version
Culture
Public Key Token

Example Strong Name:
EXESample, Version=1.0.4203.24068,
Culture=neutral, PublicKeyToken=2a79b79e3c411f38

Most of the time PublicKeyToken=null
PRIVET KEY SIGNING

Signed code is based on

- Private Key - 1024 bit
- Signed Hash of Code

Identify and Verify the Author
IT CAN’T BE THAT EZ

NO  YES
Smoothed text output:

Signed code is based on:
- Private Key - 1024 bit
- Signed Hash of Code

Signed Code Checking is off by default.
Public Key Token = b77a5c561934e089

Public Key Token = 683127632be2c302
FAKE SIGNED DLL

Turn Key Checking ON

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\.NETFramework] "AllowStrongNameBypass"=dword:00000000
FAKE SIGN DLL/EXE

Y U NO Check

Y ASK 4 PASSWORD
What is the GAC?

How to access the GAC?

Attacking from the GAC?
<table>
<thead>
<tr>
<th>Assembly Name</th>
<th>Version</th>
<th>Cul...</th>
<th>Public Key Key Token</th>
<th>Proces...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft.SqlServer</td>
<td>10.0.0.0</td>
<td></td>
<td>89845dcd8080cc91</td>
<td>MSIL</td>
</tr>
<tr>
<td>Microsoft.StdFormat</td>
<td>7.0.330.</td>
<td></td>
<td>b03f5f7f11d50a3a</td>
<td>MSIL</td>
</tr>
<tr>
<td>Microsoft.Synchronization</td>
<td>1.0.0.0</td>
<td></td>
<td>89845dcd8080cc91</td>
<td>MSIL</td>
</tr>
<tr>
<td>Microsoft.Synchronization</td>
<td>1.0.0.0</td>
<td></td>
<td>89845dcd8080cc91</td>
<td>MSIL</td>
</tr>
<tr>
<td>Microsoft.Synchronization</td>
<td>3.5.0.0</td>
<td></td>
<td>89845dcd8080cc91</td>
<td>MSIL</td>
</tr>
<tr>
<td>Microsoft.Tpm</td>
<td>6.1.0.0</td>
<td></td>
<td>31bf3856ad364e35</td>
<td>MSIL</td>
</tr>
<tr>
<td>Microsoft.Tpm.Res</td>
<td>6.1.0.0</td>
<td>en</td>
<td>31bf3856ad364e35</td>
<td>MSIL</td>
</tr>
<tr>
<td>Microsoft.Transacti...</td>
<td>3.0.0.0</td>
<td></td>
<td>b03f5f7f11d50a3a</td>
<td>MSIL</td>
</tr>
<tr>
<td>Microsoft.Transacti...</td>
<td>3.0.0.0</td>
<td></td>
<td>b03f5f7f11d50a3a</td>
<td>x86</td>
</tr>
<tr>
<td>Microsoft.Transacti...</td>
<td>3.0.0.0</td>
<td></td>
<td>b03f5f7f11d50a3a</td>
<td>AMD64</td>
</tr>
<tr>
<td>Microsoft.Vbe.Interop</td>
<td>12.0.0.0</td>
<td></td>
<td>71e9bce111e9429c</td>
<td></td>
</tr>
<tr>
<td>Microsoft.Vbe.Interop</td>
<td>11.0.0.0</td>
<td></td>
<td>71e9bce111e9429c</td>
<td></td>
</tr>
</tbody>
</table>
GLOBAL ASSEMBLY CASH
C:\Windows\assembly\
GLOBAL ASSEMBLY CASH

GAC

\GAC  –  Installed/Sandbox

\GAC_32  –  32bit-(x86)

\GAC_64  –  64bit-(x64)

\GAC_MSIL  –  MSIL(ANY)
NATIVE IMAGE(NI)

VER 1.1 - is dead 😊

GAC

VER 3.0 - is dead 😊

VER 2.0 & 3.5

-nativeImages_v2.0.50727_32
-nativeImages_v2.0.50727_64

VER 4.0

-nativeImages_v4.0.30319_32
-nativeImages_v4.0.30319_64
GAC
So much GAC!!!!!!!!!!!!!!!!!!!!!
ATTACK THE GAC
ATTACK FROM THE GAC
ATTACK FROM THE GAC
ATTACK THE GAC

1. Delete the Native Image

2. Replace File in GAC

3. Hack Target from GAC
ATTACK FROM THE GAC
ATTACK FROM FRAMEWORK

.NET FRAMEWORK
ATTACK VECTOR

ASM THE OLD IS NEW

Shell Code - ASM
UNmanaged
NO .NET Security

Execute ASM Attack with Unmanaged Calls or Reflection
THE OLD IS NEW AGAIN
ASM-SHELLS

- The Power of ASM
  - Attack from a lower level
  - Brake the “safe” security
  - Attack the Runtime
  - ASM-Shells… 😊 shells…
VISUAL STUDIO
Exploit – Run arbitrary code

First noted in 2004

Demo
PowerShell - Matrix

Get developer Keys
Attack the SVN & DB

Virus Malware
Attacking/Cracking

IN MEM |||| ON DISK
ATTACKING .NET APPLICATIONS: AT RUNTIME
WHY AT RUNTIME

Hacks
Cracks
Malware
Backdoors

SECURITY SYSTEMS

PROCESS
Inject at Runtime

GrayDragon

C++

.NET

C++

.GrayDragon

 DLL

.DLL

 DLL
Inject At Runtime
Inject At Runtime

PROCESS
ATTACKING APPS

- Gain Full Access
- Reverse Engineer
- Attack (in MEM)
- Take out the “Security”
- Control the Program
Hacking .NET Application: A Runtime Attack

Control the Runtime
Control the Application
DEMO: GOD MODE

Inject and Control
SO 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?
What is the security?
Any Malware?
Threat Level?
LOOK INSIDE

Take Control
Don't be helpless
Know you Threats
DON’T

Keys
Crypto
DB
BackDoors
MalWare
Good Code

LOOK

Passwords
Technology
Weak Spots
Data Leaks
Reg Code
Bad Code
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
The Crypto is

- DES 64
- Triple DES 192
- Rijndael AES 256
- Home MIX (secure/unsecure)
MALWARE T1M3 CRAFT WR1VAR
So your malware

How do you hide
MALWARE T1M3

Protection (Shell Crypto)
Attack (Unmanaged Calls)
Protection (Obfuscated Code)
Fake (Signed DLL Protection)
Intelligent names
Code style
Don’t use loops
Don’t use one area for your Vars
Access the normal program

Call back into your target
Use Timers
Link to Events
Spread out your Vars and Code
MALWARE FIGHT

Protect Me! 2010

Androsa FileProtector
```csharp
this.filesToAdd = new List<string>();
base..ctor();
this.InitializeComponent();
this.Text = this.AssemblyTitle + " " + this.AssemblyCultureName;
if ((int)par.Length == 1)
{
    if (par[0].Contains("en"))
    {
        this.langParEn = 1;
    }
}
```
private void x03a69b6bf165c508c()
{
    var arg_AA_0 = this.xef9c50c23fdde0e7;
    object[] array = new object[][[6];
    array[0] = this.x991baaf3e2f1814.getTranslation("meijfaagfhgaog", 127490266));
    array[1] = string.Intern(x1110bdd110cdcea4._d57
    array[2] = box(System.Int32, this.xe25232a1a3e32
    array[3] = string.Intern(x1110bdd110cdcea4._d57
    array[4] = box(System.Int32, this.xe25232a1a3e32
    array[5] = string.Intern(x1110bdd110cdcea4._d57
    arg_AA_0.Text = string.C
}
Good Crypto
Salt & VI
Encrypted Pass

Password SHA512
Custom Crypto LIB’s
Possible Back Door

Protect Me! 2010

Good Crypto
Salt & VI
Encrypted Pass

Androsa FileProtector

0b$cur17y

Androsa FileProtector
Version 1.4.4
Copyright @ AndrosaSoft 2009
MALWARE FIGHT

DEMO
PROTECTION FOR WHO?

Ob fu$ ca73
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 & Segment Risk
WHAT M$ DID WRONG

MixModeCode – Bad for security
∑ This allows ASM\C++\C 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/…/ILASLM/ILDASM
FIN
HACKER VS ATTACKER
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
101 - Recon

EHSHELL .NET Framework
Ver 3.5

Un-Obfuscated

Crash Reporting Watson

Coded in C#