Code Forensics

- Computer Architecture
- Microsoft .NET.
- Obfuscation.

Author: Prof Bill Buchanan
Web: buchananweb.co.uk

Code Forensics

Introduction

Author: Prof Bill Buchanan

Payload Analysis for Code Execution
Capture signature and Create detectors

Static Analysis of Source Code
View code.

Analysis of Network Behaviour of Executable Program
Sense operation (CPU, Memory, etc)

Reverse Engineering
Regenerate code

Focus of Presentation

Code review/debug
Find bugs, validation and evaluation.

Analysis of Host Behaviour of Executable Program
Sense operation (Network connections, Communication protocol)

Damage Analysis
View possible damage.

Author: Prof Bill Buchanan

Why Code Forensics?

Code Review on the code. Debug/walk through review.

Code Recovery. Original code lost, and organisation only has the EXE file.

Damage Analysis. Will the code do damage to the system/organisation?

Code metrics. Evaluation of the running of the code.

Debugging for bugs

Code Footprint. Analysis of the network and host footprint.

Legal Challenge. Analysis of where liability lies.

Data leakage/spying analysis. Is the program stealing or leaking data?

IP Cracking. Reverse engineering of competitor to release operation.

Code Optimisation. Understanding bottlenecks and inefficient

Vulnerability Analysis. Where are the weak/stress points?

Author: Prof Bill Buchanan

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s;

    cout << "Enter your name>>" << endl;
     cin >> s;

    cout << "Your name is " + s;
     cin >> s;

    return(0);
}
```

C++

```
namespace code_forensics_ex01
{
    class Program
    {
        static void Main(string[] args)
        {
            string name = "";
            Console.Write("Enter your name>>");
            name=Console.ReadLine();
            Console.Write("Name: " + name);
        }
    }
}
```

Csharp

```
mov   ecx, DWORD PTR _this$[ebp]
mov   edx, DWORD PTR [ecx]
mov   ecx, DWORD PTR _this$[ebp]
add   ecx, DWORD PTR [edx+4]
test  eax, eax
je    SHORT $LN10@Ipfx
```

Intel x86 code

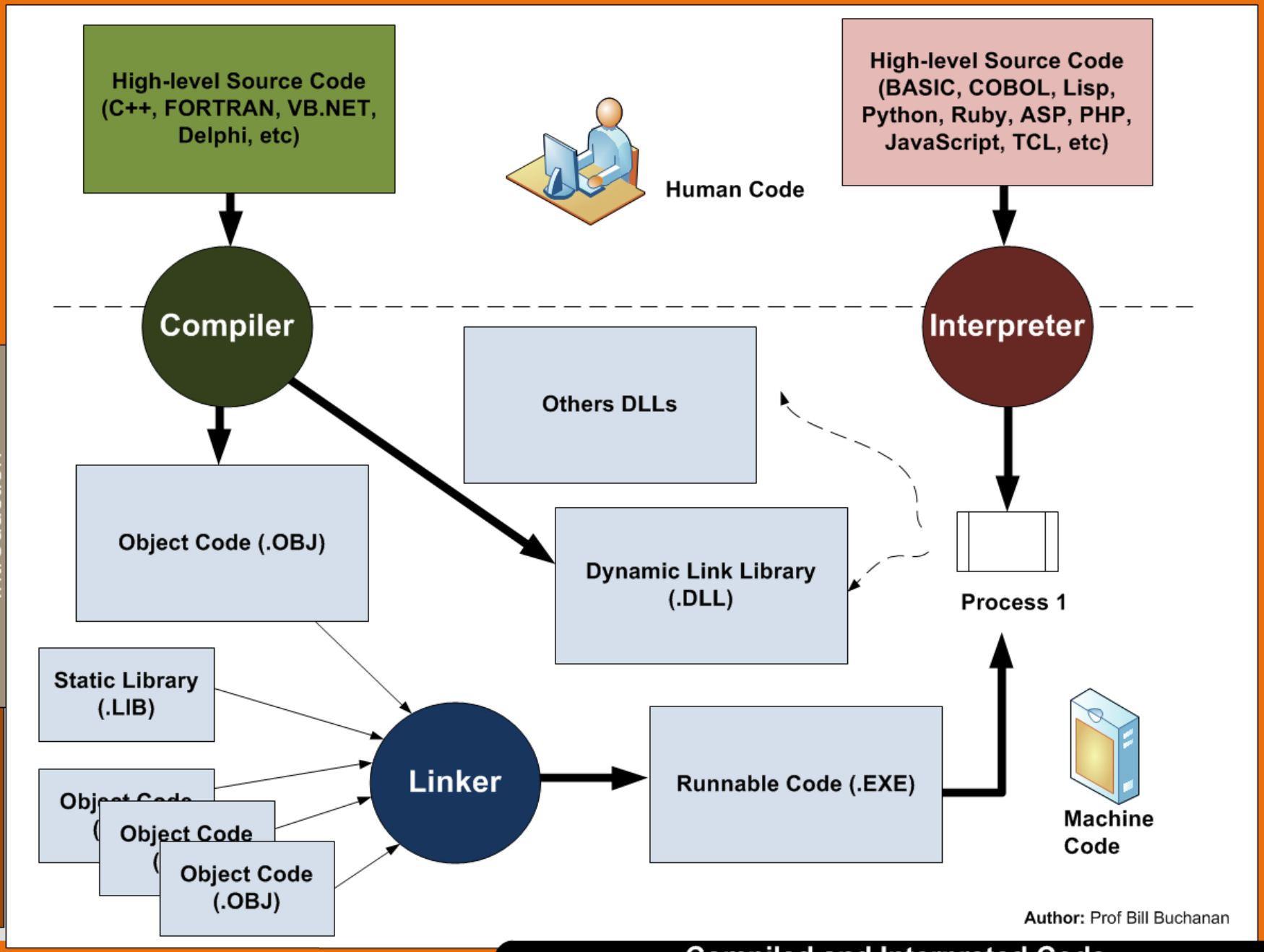
```
mov   ip, sp
stmfd sp!, {fp, ip, lr, pc}
sub   fp, ip, #4
sub   sp, sp, #4
mov   r0, #3
bl    factorial
mov   r3, r0
str   r3, [fp, #-16]
ldr   r0, .L4
```

ARM code

```
.method private hidebysig static void  Main(string[] args) cil managed
{
  .entrypoint
  // Code size       42 (0x2a)
  .maxstack  2
  .locals init ([0] string name)
  IL_0000:  nop
  IL_0001:  ldstr      ""
  IL_0006:  stloc.0
  IL_0007:  ldstr      "Enter your name>>"
  IL_000c:  call       void [mscorlib]System.Console::Write(string)
  IL_0011:  nop
  IL_0012:  call       string [mscorlib]System.Console::ReadLine()
  IL_0017:  stloc.0
  IL_0018:  ldstr      "Name: "
  IL_001d:  ldloc.0
  IL_001e:  call       string [mscorlib]System.String::Concat(string,
                                                              string)
  IL_0023:  call       void [mscorlib]System.Console::Write(string)
  IL_0028:  nop
  IL_0029:  ret
} // end of method Program::Main
```

Microsoft Intermediate Language Code (MSIL)

High-level and low level code



Compiled and Interpreted Code

Code Forensics

x86, x64, and ARM Architecture

Author: Prof Bill Buchanan

ARM Architecture

OS: Android, Chrome OS, Fedora, Ubuntu
MAC OS: iOS (iPhone, iPad)
Future: Windows 8

x86 Architecture (32-bit data bus)

OS: DOS, Windows, Linux, BSD, Solaris, Mac OS X

x64 Architecture (64-bit data bus)

OS: Windows 7 Linux, Mac OS X

Author: Prof Bill Buchanan

Intel Xeon

CPU: 1.06 GHz - 3.46 GHz
Cores: 2
L2 Cache: 256MB
(L3 cache: 4-8MB)
Power: 34-75W
Throughput: 2.5GT/s

Intel i7

CPU: 1.66-3.3GHz
Cores: 4/6
L2 Cache: 4/6x256MB
(L3 cache: 12MB)
Power: 17-95W
Throughput: 6.4GT/s

Intel i5

CPU: 1-3.8GHz
Cores: 1/4/6/8
L2 Cache: 256KB-12MB
(L3 cache: 4-16MB)
Power: 16-165W
Throughput: 6.4GT/s

Intel i3

CPU: 800MHz-2GHz
Cores: 2
L2 Cache: 256MB
(L3 cache: 3-4MB)
Power: 34-75W

CPU: 1-3.45GHz
Cores: 2/4
L2 Cache: 256MB
(L3 cache: 4-8MB)
Power: 17-95W
Throughput: 2.5GT/s

Intel Atom

CPU: 800MHz-2GHz
Cores: 1 or 2
Cache: 512K/1MB
Power: 13W

ARM Cortez A15

CPU: 1-1.5GHz (2.5GHz)
Cores: 1-4
L1 Cache: 32KB
(L2 Cache: 4MB)
Power: ?W
Android

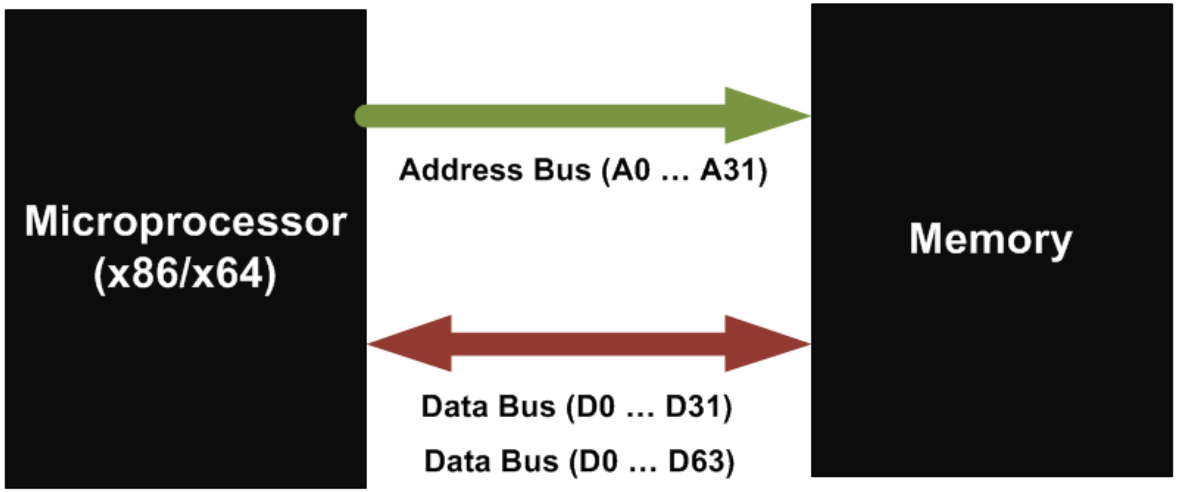
ARM Cortez A8

CPU: 600MHz
Cores: 1
L1 Cache: 32KB
(L2 Cache: 256KB)
Power: 0.3W
iPhone 3GS

Cortex
Low-Power Leadership from ARM

Author: Prof Bill Buchanan

Processor Types



Addressing: 32 bits - > 2^{32} memory locations -> 4GB
[with additional lines -> 64 GB]

Single core
(Pentium)

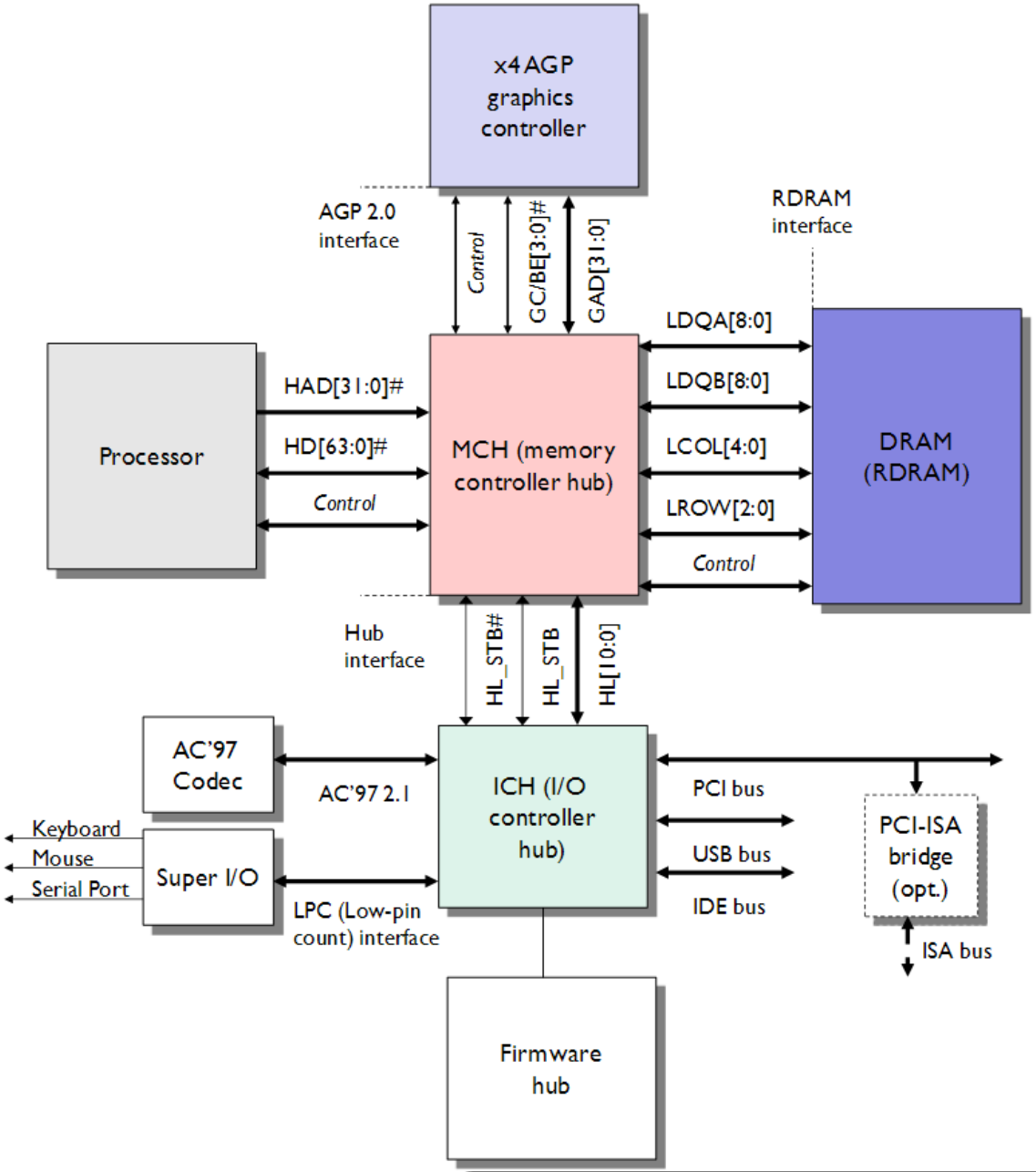
Dual core (i3)

Quad core (i5)

Hexa core (i7)

Octa core (Xeon)

Author: Prof Bill Buchanan



Author: Prof Bill Buchanan

Microprocessor package

Microprocessor

Address bus

Data bus

On-chip cache controller

Data bus

Cache tag

First-level cache (SRAM)

If cache miss, controller loads from DRAM (but the controller tries to get it right the next time)

Address bus

Data bus

On-board cache controller

Address bus

Data bus

DRAM (main memory)

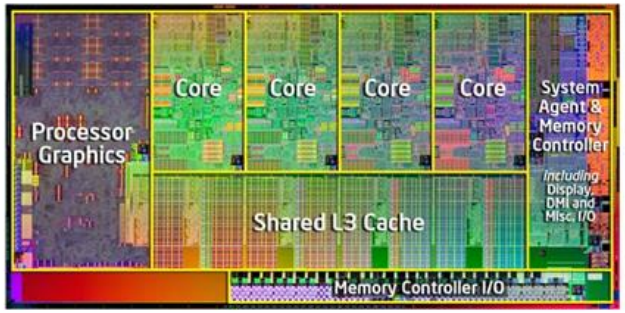
Data bus

Cache tag

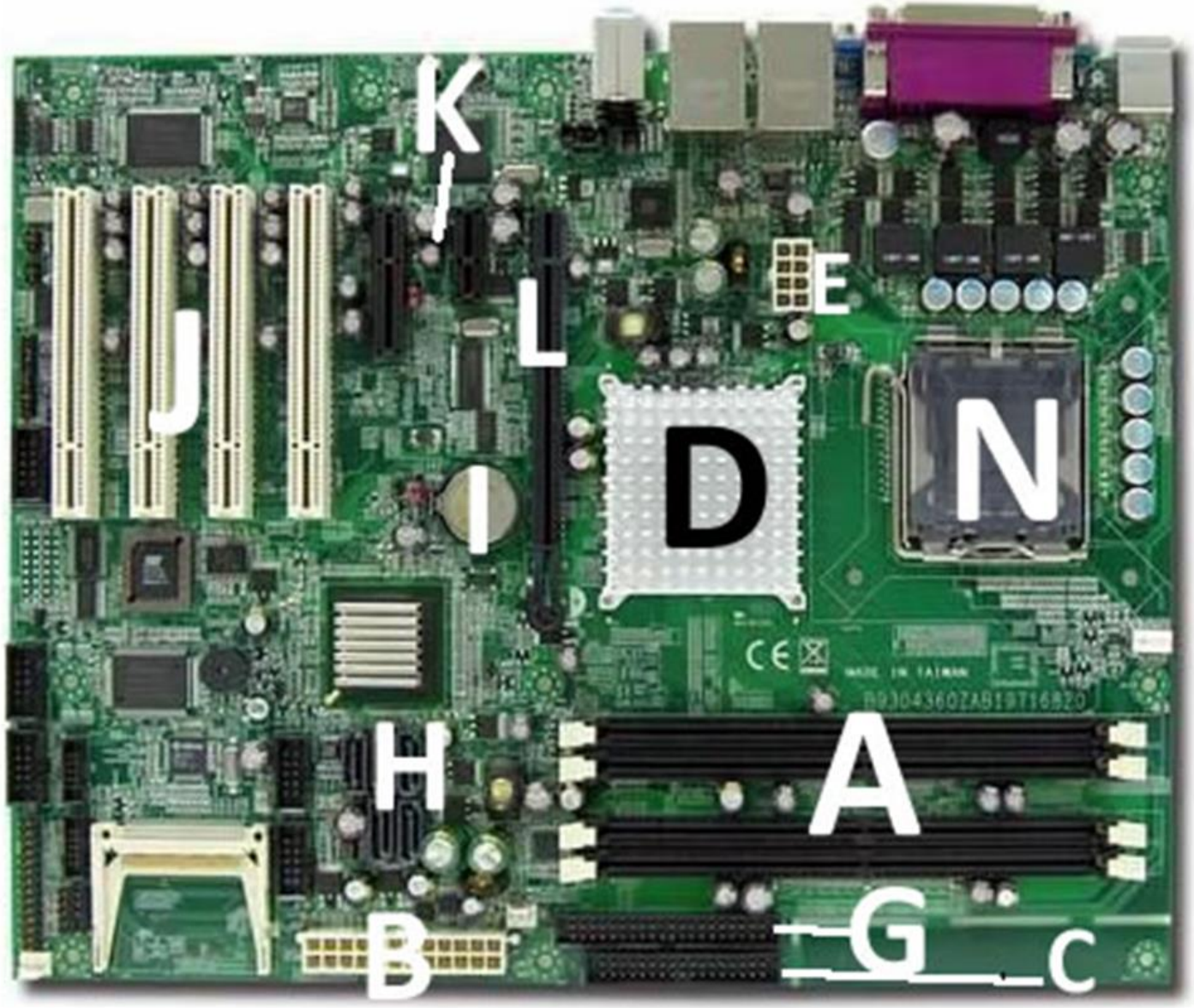
Second-level cache (SRAM)

If cache hit, processor loads from second-level cache, rather than from DRAM

Controller loads from DRAM into cache



Author: Prof Bill Buchanan



Author: Prof Bill Buchanan

Example Motherboard

Code Forensics

Code Types

Author: Prof Bill Buchanan

Code Forensics

Code Types

C++

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s;

    cout << "Enter your name>>" << endl;
     cin >> s;

    cout << "Your name is " + s;
     cin >> s;

    return(0);
}
```

Machine Code (x86)
File: OBJ

Machine Code (x86)
File: EXE

```
mov   ecx, DWORD PTR _this$[ebp]
mov   edx, DWORD PTR [ecx]
mov   ecx, DWORD PTR _this$[ebp]
add   ecx, DWORD PTR [edx+4]
test  eax, eax
je    SHORT $LN10@Ipfx
```

Hardware

Advantages: Runs directly on the hardware, Fast Code, Optimized for hardware, difficult to reverse engineer

Disadvantages: Hardware specific, Unmanaged code, difficult to debug as link to code broken

Author: Prof Bill Buchanan

C#, VB.NET

MSIL (Microsoft Intermediate Language)

.NET Runtime

Hardware

```
namespace code_forensics_ex01
{
    class Program
    {
        static void Main(string[] args)
        {
            string name = "";
            Console.Write("Enter your name>>");
            name=Console.ReadLine();
            Console.Write("Name: " + name);
        }
    }
}
```

```
.method private hidebysig static void  Main(string[] args) cil managed
{
  .entrypoint
  // Code size       42 (0x2a)
  .maxstack  2
  .locals init ([0] string name)
  IL_0000:  nop
  IL_0001:  ldstr      ""
  IL_0006:  stloc.0
  IL_0007:  ldstr      "Enter your name>>"
  IL_000c:  call       void [mscorlib]System.Console::Write(string)
  IL_0011:  nop
  IL_0012:  call       string [mscorlib]System.Console::ReadLine()
  IL_0017:  stloc.0
  IL_0018:  ldstr      "Name: "
  IL_001d:  ldloc.0
  IL_001e:  call       string [mscorlib]System.String::Concat(string,
                                                              string)
  IL_0023:  call       void [mscorlib]System.Console::Write(string)
  IL_0028:  nop
  IL_0029:  ret
} // end of method Program::Main
```

Advantages: Managed code, Compatible with a range of hardware, Language independent, Isolated code, Easy debugging as code can be matched to intermediate code

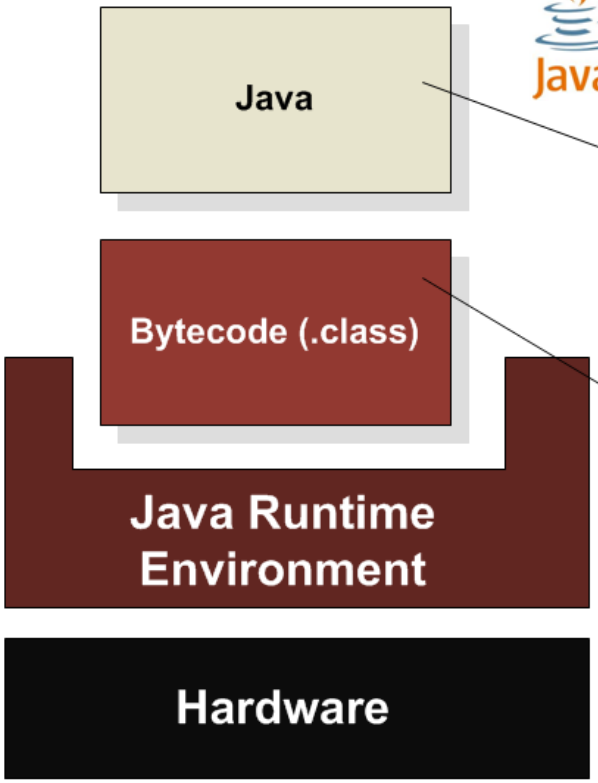
Disadvantages: Fairly slow, Versions support different features, Easy to reverse engineer

Author: Prof Bill Buchanan

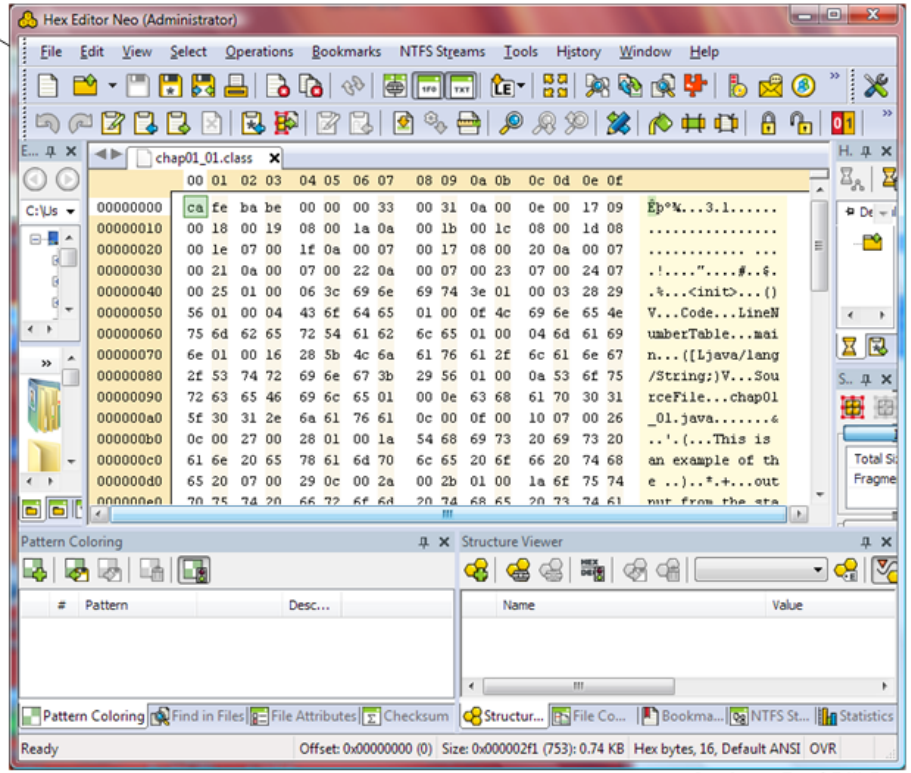
Running in a Framework (Microsoft .NET)

Code Forensics

Code Types




```
public class chap01_01
{
     public static void main(String[] args)
   {
   int i;
     i=10;
     System.out.println("This is an example of the ");
     System.out.println("output from the standalone");
     System.out.println("program");
     System.out.println("The value of i is " + i);
     }
}
```

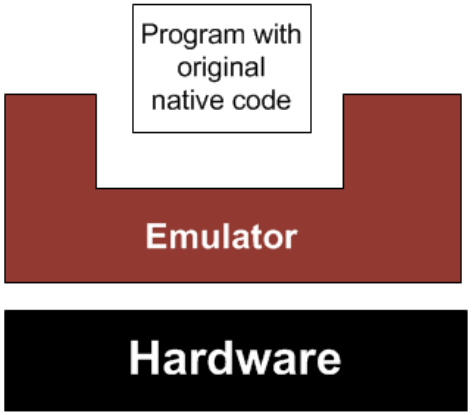


✓ **Advantages:** Managed code, Compatible with a range of hardware, Language independent, Isolated code, Easy debugging as code can be matched to intermediate code

✗ **Disadvantages:** Fairly slow, Versions support different features, Easy to reverse engineer

Author: Prof Bill Buchanan

Running in a Framework (Java)

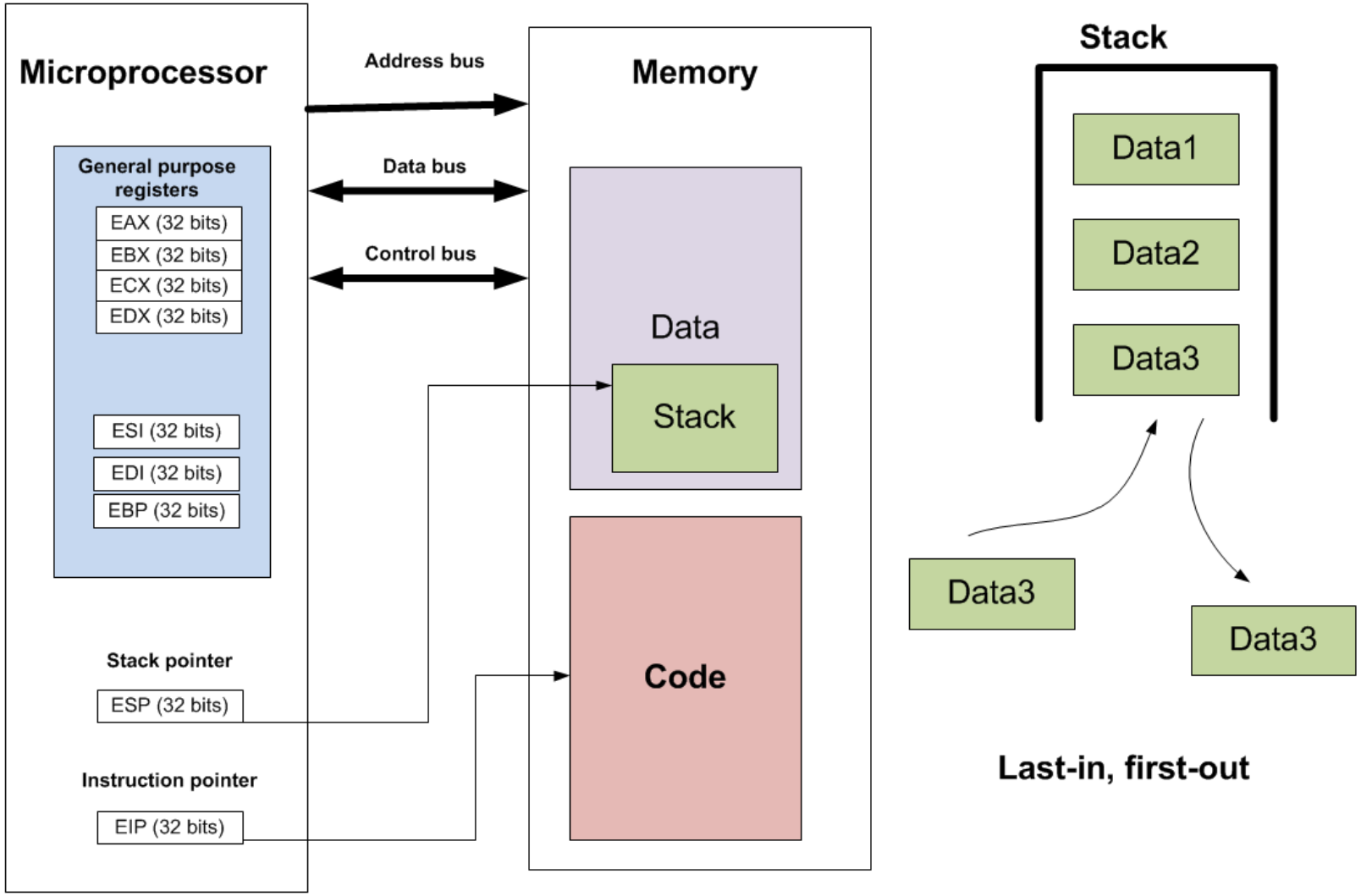


Author: Prof Bill Buchanan

Code Forensics

Machine Code Analysis

Author: Prof Bill Buchanan



Von Neumann Architecture

code_forensics_ex01 (Debugging) - Microsoft Visual Studio (Administrator)

File Edit View Project Build Debug Team Data Tools VMware Test Analyze Window Help

Memory

Disassembly

```
009D70E0  lea     ecx,[ebp-34h]
009D70E3  call    std::basic_string<char,std:
009D70E8  mov     dword ptr [ebp-4],0

    cout << "Enter your name>>" << endl;
009D70EF  mov     esi,esp
009D70F1  mov     eax,dword ptr [__imp_std::e
009D70F6  push    eax
009D70F7  push    offset string
009D70FC  mov     ecx,dword ptr
009D7102  push    ecx
009D7103  call    std::operator<
009D7108  add     esp,8
```

```
    string s;

    cout << "Enter your name>>" << endl;
    cin >> s;
```

Registers

```
EAX = 002CF8D4 EBX = 7FFD7000 ECX = 27F2B53C
EDX = 002CF400 ESI = 002CF7C4 EDI = 002CF8FC
EIP = 009D70F1 ESP = 002CF7C4 EBP = 002CF908
EFL = 00000246
```

Stack

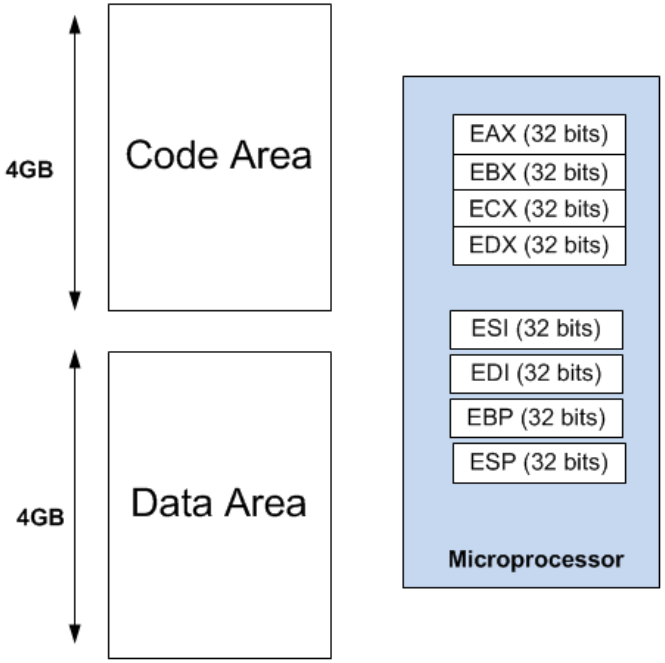
Name	Lang
code_forensics_cplus03.exe!main() Line 9 + 0x2 bytes	C++
code_forensics_cplus03.exe!__tmainCRTStartup() Line 555 + 0x19 bytes	C
code_forensics_cplus03.exe!mainCRTStartup() Line 371	C
kernel32.dll!75afd0e9()	
[Frames below may be incor	
ntdll.dll!771416c3()	
ntdll.dll!771416f6()	

Ready

Von Neumann Architecture

```
int _tmain(int argc, _TCHAR* argv[])
{
00951380  push        ebp
00951381  mov         ebp,esp
00951383  sub         esp,0E4h
00951389  push        ebx
0095138A  push        esi
0095138B  push        edi
0095138C  lea         edi,[ebp-0E4h]
00951392  mov         ecx,39h
00951397  mov         eax,0CCCCCCCCh
0095139C  rep stos    dword ptr es:[edi]

int a=1;
0095139E  mov         dword ptr [a],1
int b=2;
009513A5  mov         dword ptr [b],2
int z;

z=a+b;
009513AC  mov         eax,dword ptr [a]
009513AF  add         eax,dword ptr [b]
009513B2  mov         dword ptr [z],eax
printf("%d",z);
009513B5  mov         esi,esp
009513B7  mov         eax,dword ptr [z]
009513BA  push        eax
009513BB  push        offset string "%d" (95573Ch)
009513C0  call        dword ptr [__imp__printf (9582B0h)]
009513C6  add         esp,8
009513C9  cmp         esi,esp
009513CB  call        @ILT+305(__RTC_CheckEsp) (951136h)
```

uP (x86 processor)



EAX - Accumulator Register
EBX - Base Register
ECX - Counter Register
EDX - Data Register
ESI - Source Index
EDI - Destination Index
EBP - Base Pointer
ESP - Stack Pointer

Author: Prof Bill Buchanan

```
int _tmain(int argc, _TCHAR* argv[])
{
00951380  push        ebp
00951381  mov         ebp,esp
00951383  sub         esp,0E4h
00951389  push        ebx
0095138A  push        esi
0095138B  push        edi
0095138C  lea         edi,[ebp-0E4h]
00951392  mov         ecx,39h
00951397  mov         eax,0CCCCCCCCh
0095139C  rep stos    dword ptr es:[edi]

int a=1;
0095139E  mov         dword ptr [a],1
int b=2;
009513A5  mov         dword ptr [b],2
int z;

z=a+b;
009513AC  mov         eax,dword ptr [a]
009513AF  add         eax,dword ptr [b]
009513B2  mov         dword ptr [z],eax
printf("%d",z);
009513B5  mov         esi,esp
009513B7  mov         eax,dword ptr [z]
009513BA  push        eax
009513BB  push        offset string "%d" (95573Ch)
009513C0  call        dword ptr [__imp__printf (9582B0h)]
009513C6  add         esp,8
009513C9  cmp         esi,esp
009513CB  call        @ILT+305(__RTC_CheckEsp) (951136h)
```

Mov	Move
Int	Interrupt
Add	Add values
Sub	Subtract values
Push	Put a value on the stack
Pop	Take it off the stack
Call	Jump to a function
Ret	Return to where called
Cmp	Compare two values
Je	Jump if equal to zero
Jne	Jump if not equal to zero
Jmp	Non-conditional jump

uP (x86 processor)

Author: Prof Bill Buchanan

```
int _tmain(int argc, _TCHAR* argv[])
{
00951380  push        ebp
00951381  mov         ebp,esp
00951383  sub         esp,0E4h
00951389  push        ebx
0095138A  push        esi
0095138B  push        edi
0095138C  lea         edi,[ebp-0E4h]
00951392  mov         ecx,39h
00951397  mov         eax,0CCCCCCCCh
0095139C  rep stos    dword ptr es:[edi]

int a=1;
0095139E  mov         dword ptr [a],1
int b=2;
009513A5  mov         dword ptr [b],2
int z;

z=a+b;
009513AC  mov         eax,dword ptr [a]
009513AF  add         eax,dword ptr [b]
009513B2  mov         dword ptr [z],eax
printf("%d",z);
009513B5  mov         esi,esp
009513B7  mov         eax,dword ptr [z]
009513BA  push        eax
009513BB  push        offset string "%d" (95573Ch)
009513C0  call        dword ptr [__imp__printf (9582B0h)]
009513C6  add         esp,8
009513C9  cmp         esi,esp
009513CB  call        @ILT+305(__RTC_CheckEsp) (951136h)
```

```
EAX = CCCCCCCC
EBX = 7FFDD000
ECX = 00000000
EDX = 00000001
ESI = 00000000
EDI = 0025F7F4
EIP = 0095139E
ESP = 0025F704
EBP = 0025F7F4
EFL = 00000202
```

```
EAX = CCCCCCCC
EBX = 7FFDD000
ECX = 00000000
EDX = 00000001
ESI = 00000000
EDI = 0025F7F4
EIP = 009513A5
ESP = 0025F704
EBP = 0025F7F4
EFL = 00000202
```

```
EAX = CCCCCCCC
EBX = 7FFDD000
ECX = 00000000
EDX = 00000001
ESI = 00000000
EDI = 0025F7F4
EIP = 009513AC
ESP = 0025F704
EBP = 0025F7F4
EFL = 00000202
```

```
EAX = 00000001
EBX = 7FFDD000
ECX = 00000000
EDX = 00000001
ESI = 00000000
EDI = 0025F7F4
EIP = 009513AF
ESP = 0025F704
EBP = 0025F7F4
EFL = 00000202
```

```
EAX = 00000003
EBX = 7FFDD000
ECX = 00000000
EDX = 00000001
ESI = 00000000
EDI = 0025F7F4
EIP = 009513B2
ESP = 0025F704
EBP = 0025F7F4
EFL = 00000206
```

```
EAX = 00000003
EBX = 7FFDD000
ECX = 00000000
EDX = 00000001
ESI = 00000000
EDI = 0025F7F4
EIP = 009513B5
ESP = 0025F704
EBP = 0025F7F4
EFL = 00000206
```

uP (x86 processor)

Author: Prof Bill Buchanan

```
int _tmain(int argc, _TCHAR* argv[])
{
00BF13D0  push        ebp
00BF13D1  mov         ebp,esp
00BF13D3  sub         esp,0E4h
00BF13D9  push        ebx
00BF13DA  push        esi
00BF13DB  push        edi
00BF13DC  lea         edi,[ebp-0E4h]
00BF13E2  mov         ecx,39h
00BF13E7  mov         eax,0CCCCCCCCh
00BF13EC  rep stos    dword ptr es:[edi]

int a=1;
00BF13EE  mov         dword ptr [a],1
int b=2;
00BF13F5  mov         dword ptr [b],2
int z;

z=add(a,b);
00BF13FC  mov         eax,dword ptr [b]
00BF13FF  push        eax
00BF1400  mov         ecx,dword ptr [a]
00BF1403  push        ecx
00BF1404  call        add (0BF1091h)
00BF1409  add         esp,8
00BF140C  mov         dword ptr [z],eax
printf("%d",z);
00BF140F  mov         esi,esp
00BF1411  mov         eax,dword ptr [z]
00BF1414  push        eax
00BF1415  push        offset string "%d" (0BF573Ch)
00BF141A  call        dword ptr [__imp__printf (0BF82B0h)]
00BF1420  add         esp,8
00BF1423  cmp         esi,esp
00BF1425  call        @ILT+310(__RTC_CheckEsp) (0BF113Bh)


     return 0;
00BF142A  xor         eax,eax
}
```

```
int add(int c,int d)
{
    return (c+d);
}

int _tmain(int argc, _TCHAR* argv[])
{
int a=1;
int b=2;
int z;

z=add(a,b);
printf("%d",z);
    return 0;
}
```

Stack
(Last-in, first out)

Push value onto the stack

```
int add(int c,int d)
{
000F1390  push        ebp
000F1391  mov         ebp,esp
000F1393  sub         esp,0C0h
000F1399  push        ebx
000F139A  push        esi
000F139B  push        edi
000F139C  lea         edi,[ebp-0C0h]
000F13A2  mov         ecx,30h
000F13A7  mov         eax,0CCCCCCCCh
000F13AC  rep stos    dword ptr es:[edi]
     return (c+d);
000F13AE  mov         eax,dword ptr [c]
000F13B1  add         eax,dword ptr [d]
}
000F13B4  pop         edi
000F13B5  pop         esi
000F13B6  pop         ebx
000F13B7  mov         esp,ebp
000F13B9  pop         ebp
000F13BA  ret
```

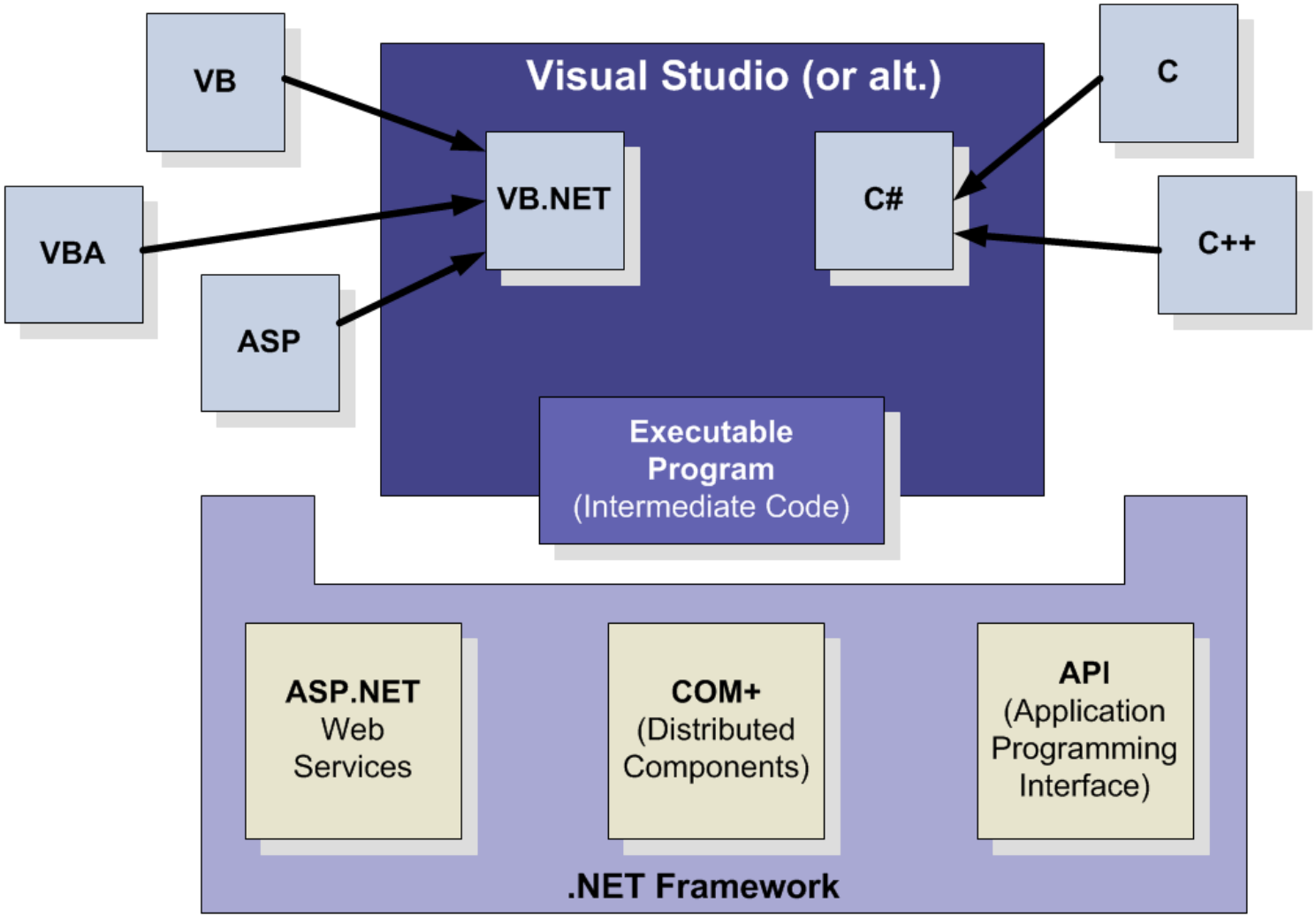
Code

Code Forensics

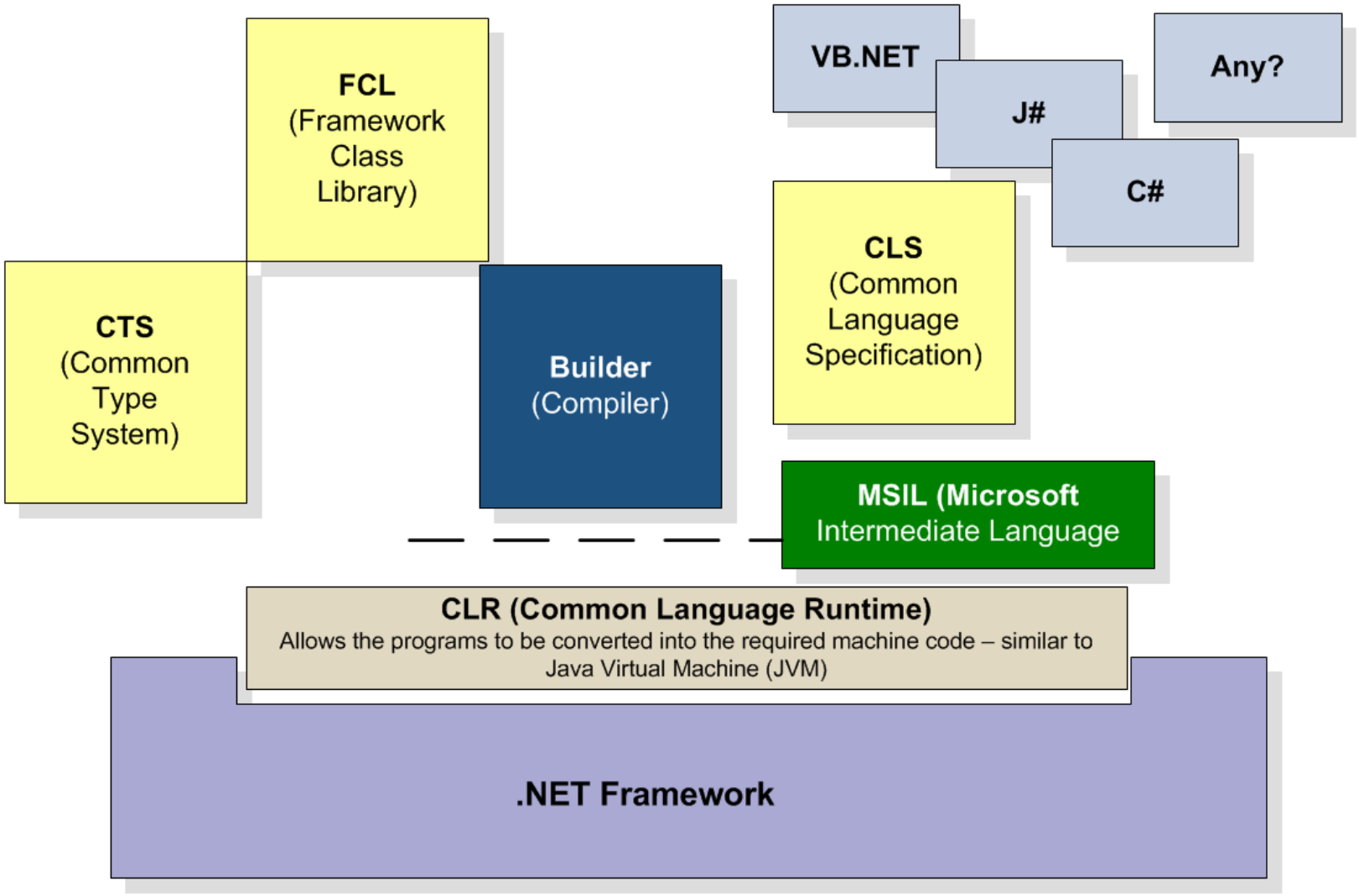
Code Forensics

Microsoft .NET

Author: Prof Bill Buchanan



Author: Prof Bill Buchanan



Author: Prof Bill Buchanan

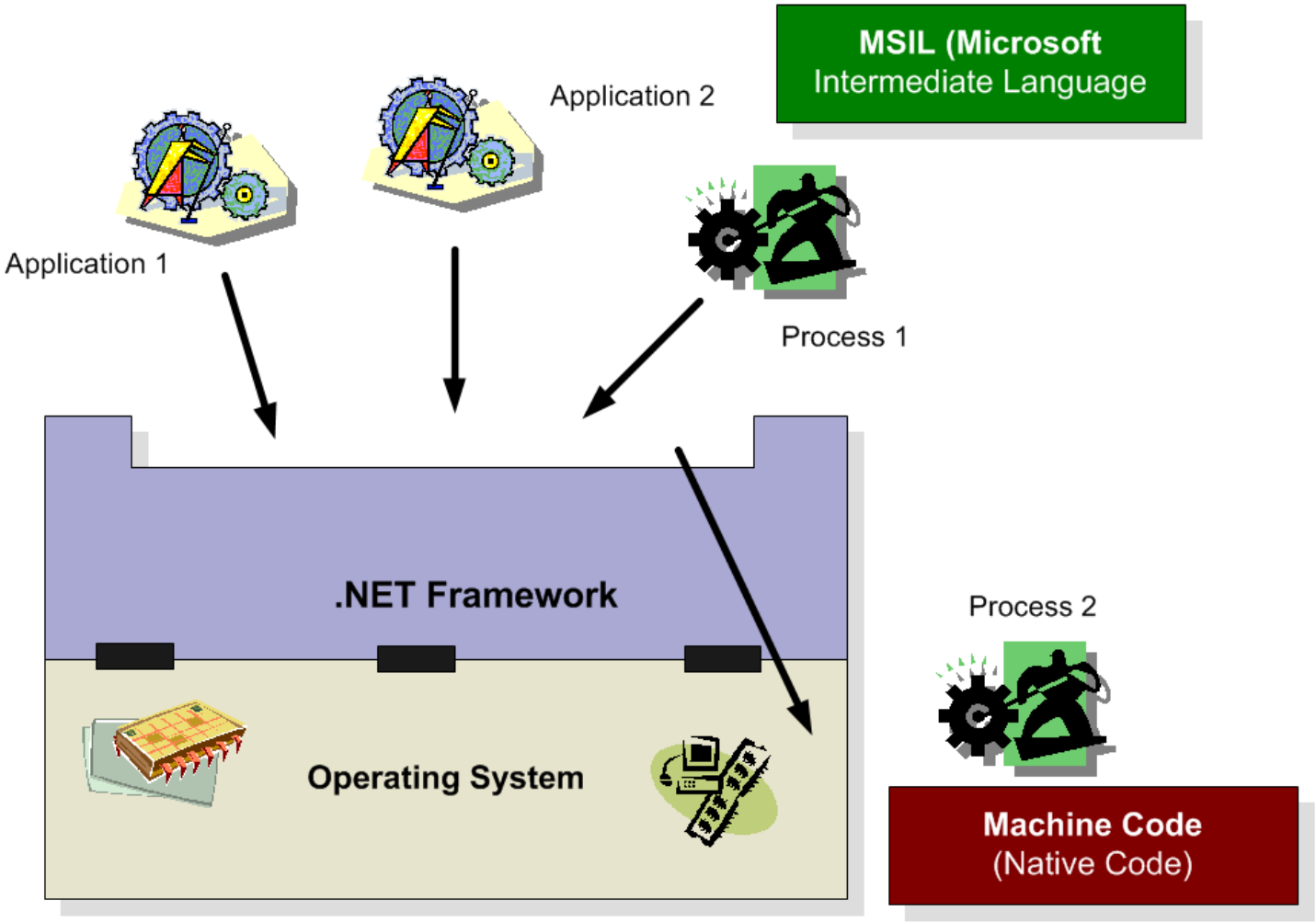
```
 Volume in drive C has no label.
 Volume Serial Number is 1A83-0D9D

 Directory of C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322

21/02/2003  08:24             7,680 Accessibility.dll
21/02/2003  06:00            98,304 alink.dll
20/02/2003  20:19            24,576 aspnet_filter.dll
20/02/2003  20:19           253,952 aspnet_isapi.dll
20/02/2003  20:19            40,960 aspnet_rc.dll
20/02/2003  20:09            77,824 CORPerfMonExt.dll
21/02/2003  11:21           626,688 cscomp.dll
21/02/2003  08:24            12,288 cscompmgd.dll
21/02/2003  08:24            33,792 CustomMarshalers.
29/07/2002  12:11           219,136 c_g18030.dll
21/02/2003  11:21           524,288 diasymreader.dll
19/03/2003  02:52           245,760 envdte.dll
20/02/2003  20:16           798,720 EventLogMessages.
20/02/2003  20:06           282,624 fusion.dll
21/02/2003  08:24             7,168 IEExecRemote.dll
21/02/2003  08:24            32,768 IEHost.dll
21/02/2003  08:24             4,608 IIEHost.dll
21/02/2003  08:25         1,564,672 mscorcfg.dll
20/02/2003  20:09            77,824 mscordbc.dll
20/02/2003  20:09           233,472 mscordbi.dll
20/02/2003  20:09            86,016 mscorie.dll
20/02/2003  20:06           311,296 mscorjit.dll
20/02/2003  20:09            98,304 mscorld.dll
21/02/2003  08:26         2,088,960 mscorlib.dll
20/02/2003  19:43           131,072 mscormmc.dll
20/02/2003  20:06            65,536 mscorpe.dll
20/02/2003  20:09           143,360 mscorrc.dll
20/02/2003  20:09            81,920 mscorsec.dll
20/02/2003  20:09            77,824 mscorsn.dll

        20/02/2003  20:07         2,494,464 mscorsvr.dll
```

.NET Program
Uses the framework for a whole host of classes: encryption, Web, networking, forms, etc.

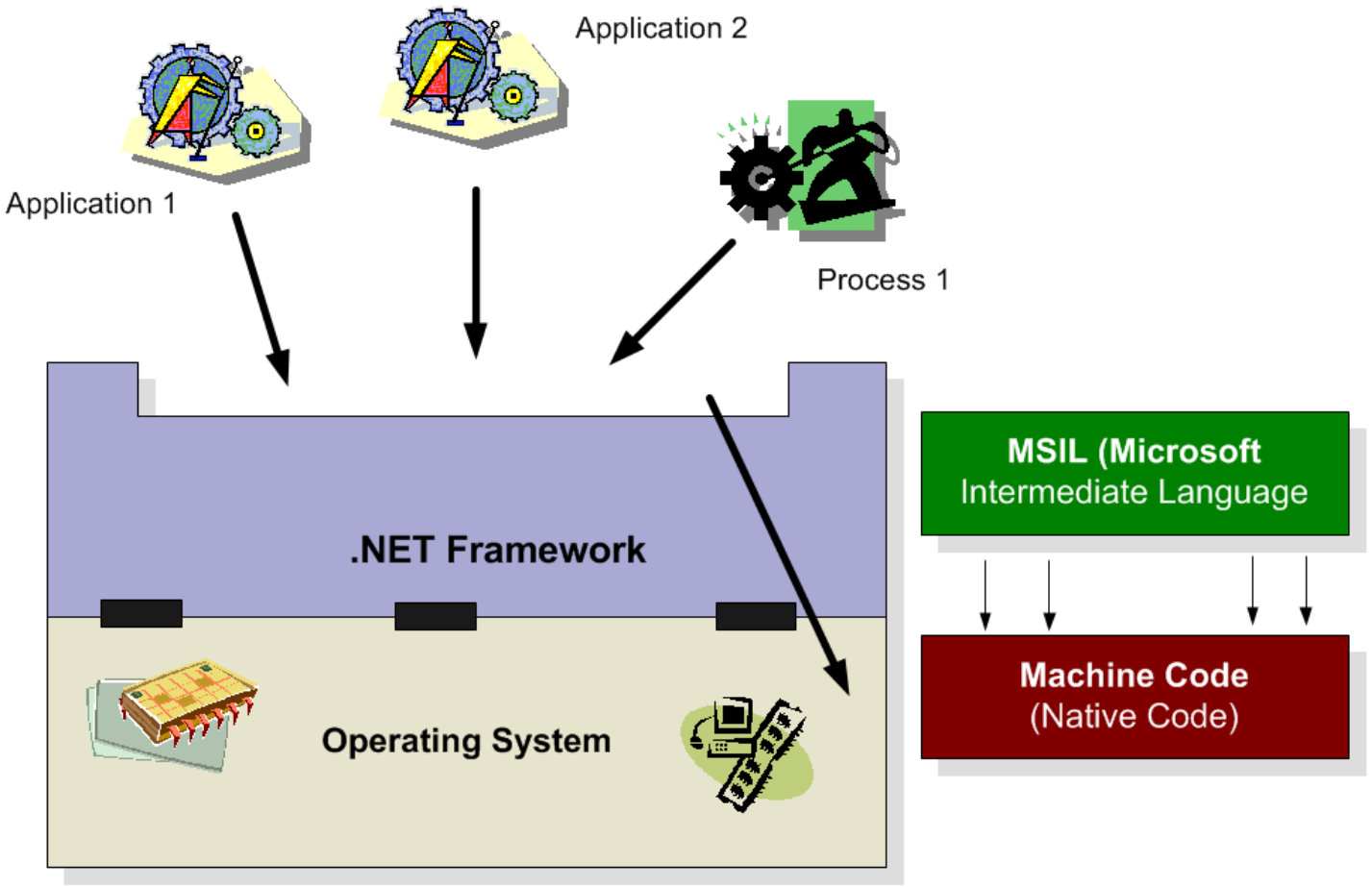
.NET Framework

Mscorlib.dll
Arrays,
File I/O,
System,
Security,
Etc.

Author: Prof Bill Buchanan



Author: Prof Bill Buchanan



Author: Prof Bill Buchanan

.NET 4.0/4.5 (April 2010/Sept 2011)

- Task Parallel Support.
- Parallel LINQ.
- Background garbage collection

.NET Framework Version 3.0 (Nov 2006, WinFX)

- Windows CardSpace
- Windows Presentation Foundation
- Windows Communication Foundation
- Windows Workflow Foundation
- LINQ (.NET 3.5)

.NET Framework Version 2.0 (Nov 2005)

- Bug fixes.
- 64-bit support.
- Language support for generics.
- New controls.
- ASP.NET components.
- Bluetooth.

.NET Framework 1.0
Jan 2002

.NET Framework 1.1 (April 2003)

- Bug fixes.
- .NET Compact Framework

Author: Prof Bill Buchanan

Code Forensics

.NET Analysis

Author: Prof Bill Buchanan

```
using System;

namespace simple
{
    class Class1
    {
        static void Main(string[] args)
        {
            string name;

            System.Console.Write("What is your name?");
            name=System.Console.ReadLine();
            System.Console.WriteLine("Hello " + name);
        }
    }
}
```

EXE

```
  .locals init ([0] string name)
  IL_0000:  ldstr      "What is your name\?"
  IL_0005:  call       void [mscorlib]System.Console::Write(string)
  IL_000a:  call       string [mscorlib]System.Console::ReadLine()
  IL_000f:  stloc.0
  IL_0010:  ldstr      "Hello "
  IL_0015:  ldloc.0
  IL_0016:  call       string [mscorlib]System.String::Concat(string,
                                                              string)
  IL_001b:  call       void [mscorlib]System.Console::WriteLine(string)
  IL_0020:  ret
} // end of method Class1::Main
```

```
F:\docs\src\simple\test>dir
 Volume in drive F has no label.
 Volume Serial Number is 2886-0553

 Directory of F:\docs\src\simple\test

25/07/2008  01:20    <DIR>          .
25/07/2008  01:20    <DIR>          ..
28/01/2007  17:06               437 simple.cs
               1 File(s)            437 bytes
               2 Dir(s)  113,418,530,816 bytes free

F:\docs\src\simple\test>csc simple.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.21022.8
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.


F:\docs\src\simple\test>dir
 Volume in drive F has no label.
 Volume Serial Number is 2886-0553

 Directory of F:\docs\src\simple\test

15/09/2008  16:37    <DIR>          .
15/09/2008  16:37    <DIR>          ..
28/01/2007  17:06               437 simple.cs
15/09/2008  16:37             4,096 simple.exe
               2 File(s)          4,533 bytes
               2 Dir(s)  113,418,526,720 bytes free

F:\docs\src\simple\test>simple
Program to determine the square root of a value.
Enter value 1:9
Square root is:3
```

Run program

```
F:\docs\src\simple\test>exemplar simple.exe > list.cs

F:\docs\src\simple\test>dir
 Volume in drive F has no label.
 Volume Serial Number is 2886-0553

 Directory of F:\docs\src\simple\test

15/09/2008  16:38    <DIR>          .
15/09/2008  16:38    <DIR>          ..
15/09/2008  16:38               451 list.cs
28/01/2007  17:06               437 simple.cs
15/09/2008  16:37             4,096 simple.exe

F:\docs\src\simple\test>type list.cs
namespace simple {
        class simple {

                [STAThread]
                private static void Main(string[] args) {
                        double local0;
                        double local1;

                        Console.WriteLine("Program to determine the
square root of a value.");
                        Console.Write("Enter value 1:");
                        local0 = Convert.ToDouble(Console.ReadLine());
                        local1 = Math.Sqrt(local0);
                        Console.WriteLine("Square root is:" + local1);
                }

                public simple() : base() {
                }
        }
}
```

Red Gate's .NET Reflector

File View Tools Help

C#

- mscorlib
- System
- System.Xml
- System.Data
- System.Web
- System.Drawing
- System.Windows.Forms
- simple
  - simple.exe
    - References
    - { } -
    - { } simple
      - simple
        - Base Types
        - Derived Types
        - .ctor()
        - Main(String[]) : Void

Disassembler

```
[STAThread]
private static void Main(string[] args)
{
    Console.WriteLine("Program to determine the square root of a value.");
    Console.Write("Enter value 1:");
    Console.WriteLine("Square root is:" + Math.Sqrt(Convert.ToDouble(Console.ReadLine())));
}
```

```
private static void Main(string[] args);
```

Declaring Type: simple.simple

Assembly: simple, Version=0.0.0.0

Author: Prof Bill Buchanan

.NET Decompiler, Java Decompiler Resources - Windows Internet Explorer

http://www.preemptive.com/decompilers/

File Edit View Favorites Tools Help

.NET Decompiler, Java Decompiler Resources

Home | Products | Free Downloads | Partners | Support | Company | News | Contact Us | PreEmptive Solutions

Search

Buy Now

Free Trial | See a Demo

index

Secure Coding Quiz

Decompilers for .NET, Decompilers for Java

What does a Java decompiler or a .NET decompiler do?

Java decompilers and .NET decompilers are designed to accept an executable such as a Java class or jar file, or a .NET exe or dll file as input, and produce a compilable source file as its result.

How good are Java decompilers and .NET decompilers?

Programs in Java or the .NET framework are easy to decompile. This is simply a reality of modern, intermediate-compiled languages. Both Java and .NET mutually share the use of expressive file syntax for delivery of executable code: bytecode in the case of Java, MSIL (Microsoft Intermediate Language) for .NET. Being much higher-level than binary machine code, the intermediate files contain identifiers and algorithms that are immediately observable and ultimately understandable.

See for yourself-- download a Java or .NET decompiler below or watch a demonstration on our decompiler and obfuscation demo page.

Should I use a Java or .NET decompiler that allows me to submit code from my browser?

Be careful. Those decompiler sites might preserve the source code of the files you submitted. It is best to download a free Java or .NET decompiler and run it locally.

How do I protect my Java or .NET code from reverse engineering?

Using an obfuscator will give you peace of mind when you deploy your applications. A good .NET obfuscator or Java obfuscator can put the reverse engineering bar back to or higher than it was before the days of intermediate languages.

Free Java Decompilers

- **Jad**: a Java decompiler written in C++. Jad is command-line only, but various GUIs are available on the download page.
- **JODE**: a powerful Java decompiler written in Java.

Free .NET Decompilers

- **Reflector for .NET**: Reflector is a .NET decompiler, featuring an excellent interface and fast, effective decompilation. Free add-ins extend Reflector's functionality, allowing for various useful operations. The FileDisassembler add-in will dump an assembly decompiled with Reflector to a set of source files that can then be searched for critical strings, or loaded, edited and recompiled.
- **Anakrino**: Another .NET decompiler, although it is becoming dated.

Protect yourself from Decompilers:

Java Obuscator- Dash O

.NET Obfuscator - Dotfuscator

Get a free evaluation copy

Internet 100%

```
c:\docs\code_forensics\jad>dir
23/10/2011  16:03                 298 chap01_01.java

c:\docs\code_forensics\jad>javac chap01_01.java

c:\docs\code_forensics\jad>dir
23/10/2011  16:50                 753 chap01_01.class
23/10/2011  16:03                 298 chap01_01.java

c:\docs\code_forensics\jad>java chap01_01
This is an example of the
output from the standalone
program
The value of i is 10

c:\docs\code_forensics\jad>jad chap01_01.class
Parsing chap01_01.class... Generating chap01_01.jad

c:\docs\code_forensics\jad>type chap01_01.jad
import java.io.PrintStream;
public class chap01_01
{
    public chap01_01()
    {
    }
    public static void main(String args[])
    {
        byte byte0 = 10;
        System.out.println("This is an example of the ");
        System.out.println("output from the standalone");
        System.out.println("program");
        System.out.println((new StringBuilder()).append("The value of i is ").ap
pend(byte0).toString());
    }
}
```

Code Forensics

Obfuscation

Author: Prof Bill Buchanan

```
#include <stdio.h> main(t,_,a)char
*a;{return!0<t?t<3?main(-79,-13,a+main(-87,1-_,
main(-86,0,a+1)+a)):1,t<_?main(t+1,_,a):3,main(-
94,-27+t,a)&&t==2?_<13? main(2,_+1,"%s %d %d\
n"):9:16:t<0?t<-72?main(_,t, "@n'+,#'/*{}w+/
w#cdnr/+,{}r/*de}+,/*{*+,/w{%+,/w#q#n+,/#{l,+,/
n{n+,/+#n+,/#\ ;#q#n+,/+k#;*+,/'r :'d*'3,}{w+K
w'K:'+}e#';dq#'l \ q#'+d'K#!/
+k#;q#'r}eKK#}w'r}eKK{nl]'/#;#q#n'){)#}w'){){nl]'/
+#n';d}rw' i;# \ ){nl]!/n{n#'; r{#w'r nc{nl]'/
#{l,+'K {rw' iK{;[{nl]'/w#q#n'wk nw' \
iwk{KK{nl]!/w{%'l##w#' i; :{nl]'/
*{q#'ld;r'}{nlwb!/*de}'c \ ;;{nl'-{}rw]'/
+,}##'*}#nc,',#nw]'/+kd'+e}+;#'rdq#w! nr'/ ')
}+}{rl#'{n' ')# \ }'+}##(!!/") :t<-
50?_==*a?putchar(31[a]):main(-
65,_,a+1):main((*a=='/')+t,_,a+1)
:0<t?main(2,2,"%s"):*a=='/'||main(0,main(-61,*a,
"!ek;dc i@bK'(q)-[w]*%n+r3#l,{}:\nuwloca-O;m
.vpbks,fxntdCeghiry"),a+1);}
```

Obfuscation

Code Forensics

Author: Prof Bill Buchanan

On the first day of Christmas,
my true love sent to me
A partridge in a pear tree.
On the second day of Christmas,
my true love sent to me
Two turtle doves,
And a partridge in a pear tree….

Elimination of all whitespace.
Use of conditional and list expression instead of the more familiar if-then-else statement and statement blocks.
A simple encoding of the poem's strings.
Encoding of multiple "functions" into the single function main

Author: Prof Bill Buchanan

Identifier Renaming

This involves renaming all the classes, methods, and fields to short names, or even non-printing names

Before

```
namespace Emulator {
        public class gen_switch {
          public ArrayList32logging59
          public ArrayList32level5Commands59
          public string32level5Name59
          public ArrayList32level69Commands59
          public string32level69Name59
          public ArrayList32level39Commands59
          public string32level39Name59
          public ArrayList32level16Commands59
          public string32level16Name59
          public ArrayList32level17Commands59
          public string32level17Name59
          public ArrayList32level25Commands59
          public string32level25Name59
```

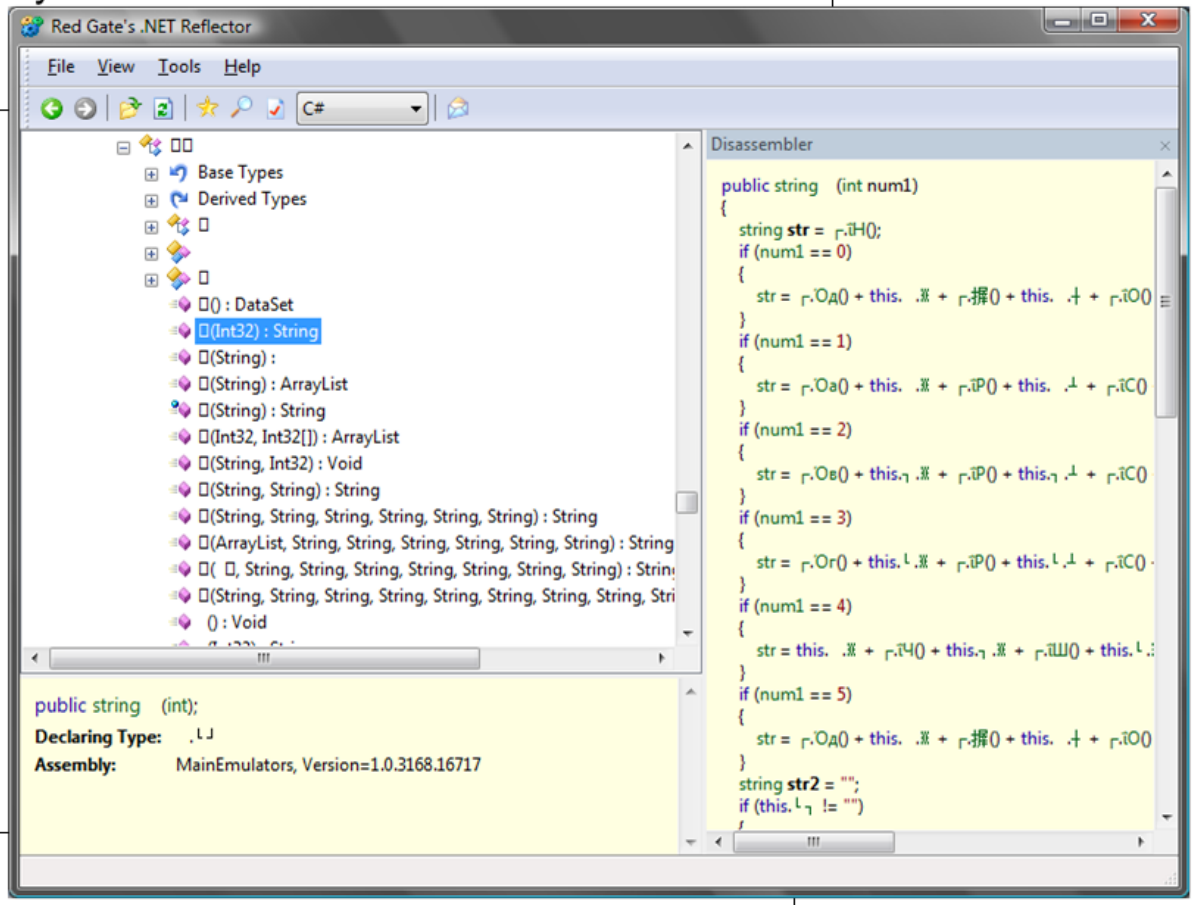
After

```
C:\netwsims>exemplar mainemulators.exe
namespace ☺ {
        public class ☺ {
                public ArrayList32☺59
                public ArrayList32☻59
                public string32☺59
                public ArrayList32♥59
                public string32☻59
                public ArrayList32♦59
                public string32♥59
                public ArrayList32♣59
                public string32♦59
                public ArrayList32♠59
                public string32♣59
                public ArrayList32♫59
```

Author: Prof Bill Buchanan

String Encryption

A standard way that many “crackers” work is to search for key strings within a program. String encryption is used to encrypt strings through an encryption method, so that they cannot be search for.



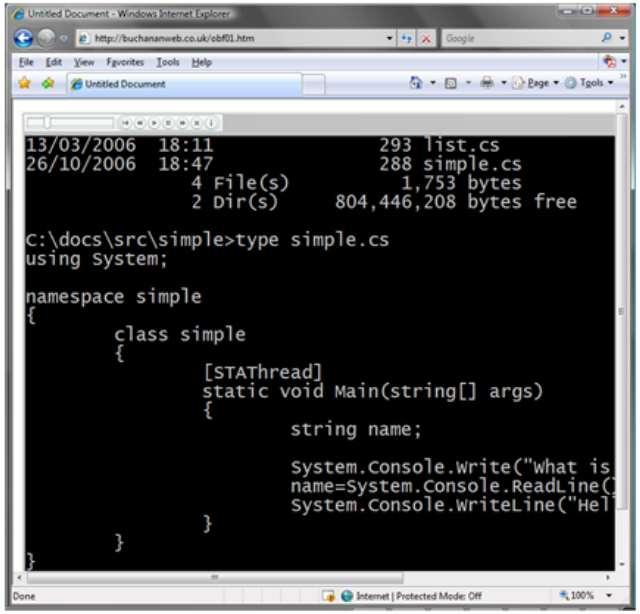
Flow obfuscation

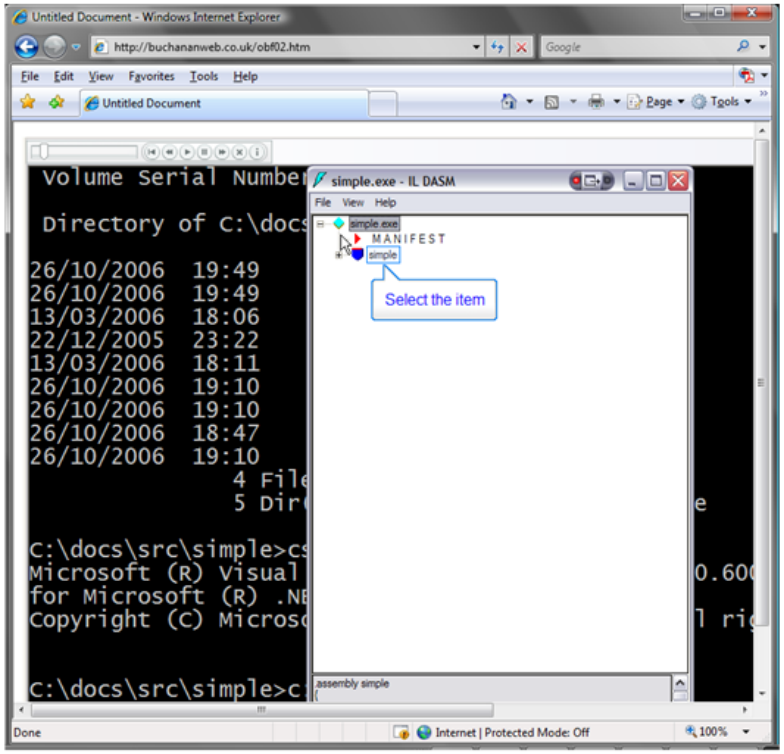
This involves scrambling the flow of the program, so that it is difficult to determine its actual operation.

Author: Prof Bill Buchanan

View demo of Obfuscation

http://buchananweb.co.uk/obf01.htm





View demo of Obfuscation2

http://buchananweb.co.uk/obf02.htm

Author: Prof Bill Buchanan

```
using System;

namespace simple
{
    class Class1
    {
        static void Main(string[] args)
        {
            string name;

            System.Console.Write("What is your name?");
            name=System.Console.ReadLine();
            System.Console.WriteLine("Hello " + name);
        }
    }
}
```

EXE

```
  .locals init ([0] string name)
  IL_0000:  ldstr      "What is your name\?"
  IL_0005:  call       void [mscorlib]System.Console::Write(string)
  IL_000a:  call       string [mscorlib]System.Console::ReadLine()
  IL_000f:  stloc.0
  IL_0010:  ldstr      "Hello "
  IL_0015:  ldloc.0
  IL_0016:  call       string [mscorlib]System.String::Concat(string,
                                                              string)
  IL_001b:  call       void [mscorlib]System.Console::WriteLine(string)
  IL_0020:  ret
} // end of method Class1::Main
```

Obfuscation

Code Forensics

Class1::Main : void(string[])

```
.method private hidebysig static void  Main(string[] args) cil managed
{
  .entrypoint
  .custom instance void [mscorlib]System.STAThreadAttribute::.ctor() = ( 01
  // Code size       33 (0x21)
  .maxstack  2
  .locals init ([0] string name)
  IL_0000:  ldstr      "What is your name\?"
  IL_0005:  call       void [mscorlib]System.Console::Write(string)
  IL_000a:  call       string [mscorlib]System.Console::ReadLine()
  IL_000f:  stloc.0
  IL_0010:  ldstr      "Hello "
  IL_0015:  ldloc.0
  IL_0016:  call       string [mscorlib]System.String::Concat(string,
                                                              string)
  IL_001b:  call       void [mscorlib]System.Console::WriteLine(string)
  IL_0020:  ret
} // end of method Class1::Main
```

Variables have been renamed

Microsoft .NET

```
) cil managed

te::.ctor() = ( 01
  .maxstack  2
  .locals init (string V_0)
  IL_0000:  ldstr      "What is your name\?"
  IL_0005:  call       void class [mscorlib]System.Console::Write(string)
  IL_000a:  call       string class [mscorlib]System.Console::ReadLine()
  IL_000f:  stloc.0
  IL_0010:  ldstr      "Hello "
  IL_0015:  ldloc.0
  IL_0016:  call       string string::Concat(string,
                                             string)
  IL_001b:  call       void class [mscorlib]System.Console::WriteLine(string
  IL_0020:  ret
} // end of method a::Main
```

Classes have been renamed

Obfuscation

Code Forensics

Class1::Main : void(string[])

```
.method private hidebysig static void  Main(string[] args) cil managed
{
  .entrypoint
  .custom instance void [mscorlib]System.STAThreadAttribute::.ctor() = ( 01
  // Code size       33 (0x21)
  .maxstack  2
  .locals init ([0] string name)
  IL_0000:  ldstr      "What is your name\?"
  IL_0005:  call       void [mscorlib]System.Console::Write(string)
  IL_000a:  call       string [mscorlib]System.Console::ReadLine()
  IL_000f:  stloc.0
  IL_0010:  ldstr      "Hello "
  IL_0015:  ldloc.0
  IL_0016:  call       string [mscorlib]System.String::Concat(string,
                                                              string)
  IL_001b:  call       void [mscorlib]System.Console::WriteLine(string)
  IL_0020:  ret
} // end of method Class1::Main
```

Variables have been renamed

Microsoft .NET

Microsoft Visual C++ Runtime Library

Runtime Error!

Program: c:\exemplar.exe

This application has requested the Runtime to terminate it in an unusual way.
Please contact the application's support team for more information.

OK

```
  .maxstack
  .locals i
  IL_0000:
  IL_0005:
  IL_000a:
  IL_000f:
  IL_0010:  ldstr      "Hello "
  IL_0015:  ldloc.0
  IL_0016:  call       string string::Concat(string,
                                             string)
  IL_001b:  call       void class [mscorlib]System.Console::WriteLine(string
  IL_0020:  ret
} // end of method a::Main
```

Classes have been renamed

Obfuscation

Code Forensics

Code Forensics

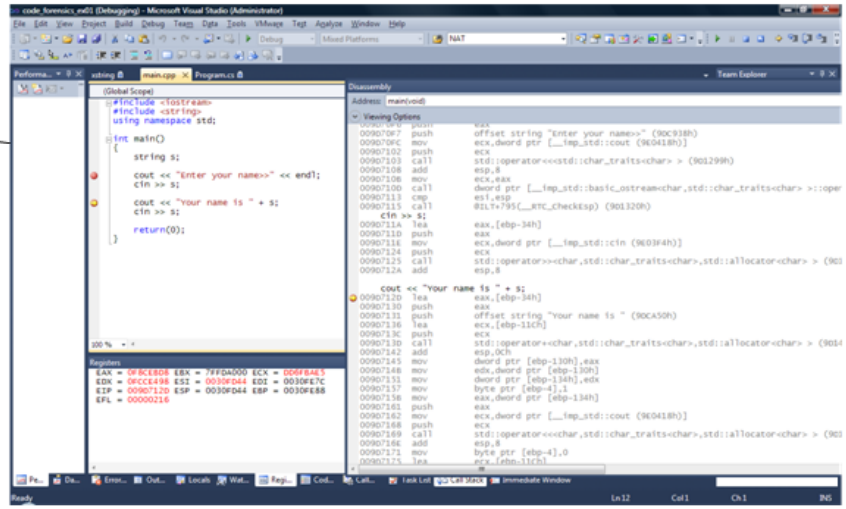
Host-based Analysis

Author: Prof Bill Buchanan

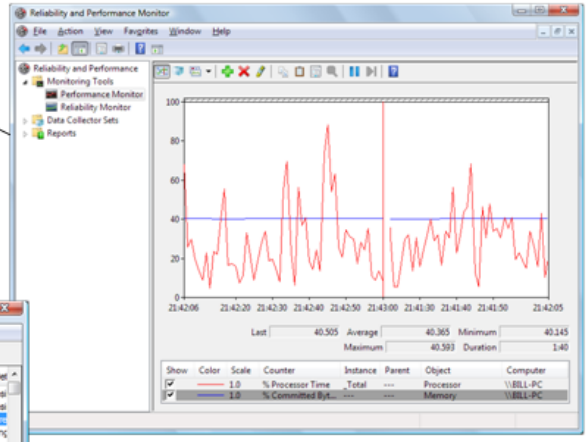
Host-based Analysis

Code Forensics

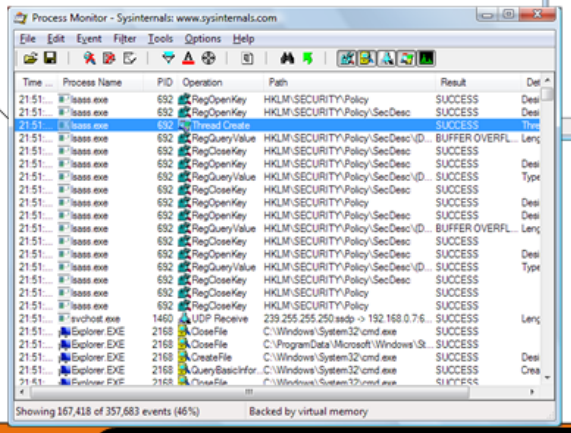
Debugging running program. Examine process, registers, etc



Dynamic analysis of the program. Examine CPU utilization, memory, etc.



File trace. Examine Windows registry, file usage, etc.

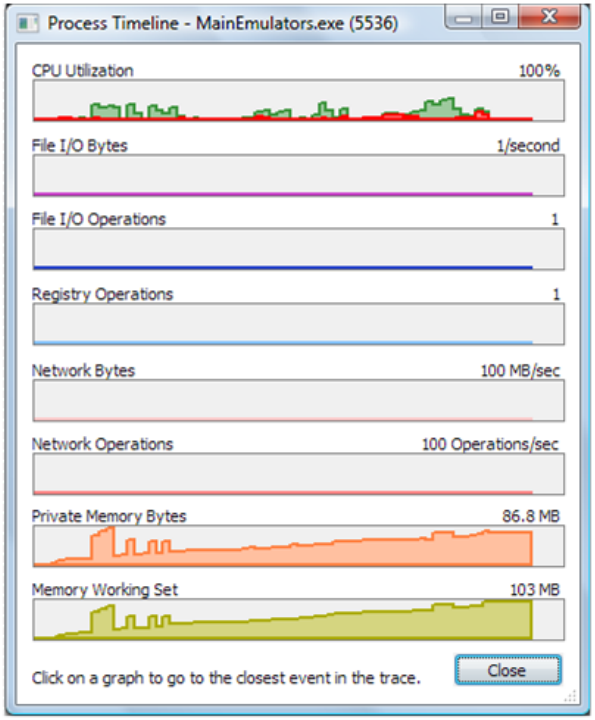


Validating files

Author: Prof Bill Buchanan

Host Analysis

Dynamic analysis of the program. Examine CPU utilization, memory, etc.



Author: Prof Bill Buchanan

Debugging running program. Examine process, registers, etc

code_forensics_ex01 (Debugging) - Microsoft Visual Studio (Administrator)

File Edit View Project Build Debug Team Data Tools VMware Test Analyze Window Help

Memory 1

Address: 0x009D70A0

Memory

Disassembly

Address: main(void)

```
009D70E0  lea     ecx,[ebp-34h]
009D70E3  call    std::basic_string<char,std:
009D70E8  mov     dword ptr [ebp-4],0

    cout << "Enter your name>>" << endl;
009D70EF  mov     esi,esp
009D70F1  mov     eax,dword ptr [__imp_std::e
009D70F6  push    eax
009D70F7  push    offset string
009D70FC  mov     ecx,dword ptr
009D7102  push    ecx
009D7103  call    std::operator<
009D7108  add     esp,8
```

Disassembly

```
string s;

cout << "Enter your name>>" << endl;
cin >> s;
```

Registers

```
EAX = 002CF8D4 EBX = 7FFD7000 ECX = 27F2B53C
EDX = 002CF400 ESI = 002CF7C4 EDI = 002CF8FC
EIP = 009D70F1 ESP = 002CF7C4 EBP = 002CF908
EFL = 00000246
```

Registers

Call Stack

Name	Lang
code_forensics_cplus03.exe!main() Line 9 + 0x2 bytes	C++
code_forensics_cplus03.exe!__tmainCRTStartup() Line 555 + 0x19 bytes	C
code_forensics_cplus03.exe!mainCRTStartup() Line 371	C
kernel32.dll!75afd0e9()	
[Frames below may be incor	
ntdll.dll!771416c3()	
ntdll.dll!77141696()	

Stack

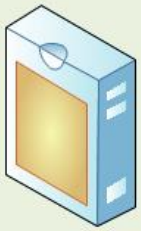
Ready

Host-based Analysis

Code Forensics

Author: Prof Bill Buchanan

File trace. Examine Windows registry, file usage, etc.



Process Monitor - Sysinternals: www.sysinternals.com

File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	D
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\System\CurrentControlSet\Control\Sessio...	REPARSE	De
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\System\CurrentControlSet\Control\Sessio...	SUCCESS	De
21:59:...	MainEmulators....	5536	RegQueryValue	HKLM\System\CurrentControlSet\Control\Sessio...	NAME NOT FOUND	Ler
21:59:...	MainEmulators....	5536	RegCloseKey	HKLM\System\CurrentControlSet\Control\Sessio...	SUCCESS	
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\System\CurrentControlSet\Control\SafeBo...	REPARSE	De
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\System\CurrentControlSet\Control\SafeBo...	NAME NOT FOUND	De
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\Software\Policies\Microsoft\Windows\Saf...	SUCCESS	De
21:59:...	MainEmulators....	5536	RegQueryValue	HKLM\SOFTWARE\Policies\Microsoft\Windows...	NAME NOT FOUND	Ler
21:59:...	MainEmulators....	5536	RegCloseKey	HKLM\SOFTWARE\Policies\Microsoft\Windows...	SUCCESS	
21:59:...	MainEmulators....	5536	RegOpenKey	HKCU\Software\Policies\Microsoft\Windows\Saf...	NAME NOT FOUND	De
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\System\CurrentControlSet\Control\Sessio...	REPARSE	De
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\System\CurrentControlSet\Control\Sessio...	SUCCESS	De
21:59:...	MainEmulators....	5536	RegQueryValue	HKLM\System\CurrentControlSet\Control\Sessio...	NAME NOT FOUND	Ler
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\SOFTWARE\Microsoft\Windows NT\Curr...	SUCCESS	De
21:59:...	MainEmulators....	5536	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\Curr...	NAME NOT FOUND	Ler
21:59:...	MainEmulators....	5536	RegCloseKey	HKLM\SOFTWARE\Microsoft\Windows NT\Curr...	SUCCESS	
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\System\Setup	SUCCESS	De
21:59:...	MainEmulators....	5536	RegQueryValue	HKLM\SYSTEM\Setup\SystemSetupInProgress	SUCCESS	Typ
21:59:...	MainEmulators....	5536	RegCloseKey	HKLM\SYSTEM\Setup	SUCCESS	
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM	SUCCESS	De
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\Software\Microsoft\Windows NT\Current...	NAME NOT FOUND	De
21:59:...	MainEmulators....	5536	RegOpenKey	HKLM\Software\Microsoft\.NETFramework\Policy	SUCCESS	De

Showing 12,676 of 512,102 events (2.4%) Backed by virtual memory

Process Monitor - Sysinternals: www.sysinternals.com

File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	D
21:59:...	MainEmulators....	5536	Process Start		SUCCESS	Par
21:59:...	MainEmulators....	5536	Thread Create		SUCCESS	Thr
21:59:...	MainEmulators....	5536	Load Image	C:\docs\src\emulators\MainEmulators.exe	SUCCESS	Ima
21:59:...	MainEmulators....	5536	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS	Ima
21:59:...	MainEmulators....	5536	CreateFile	C:\Windows\Prefetch\MAINEMULATORS.EXE-...	SUCCESS	De
21:59:...	MainEmulators....	5536	QueryStandardI...	C:\Windows\Prefetch\MAINEMULATORS.EXE-...	SUCCESS	Allc
21:59:...	MainEmulators....	5536	QueryStandardI...	C:\Windows\Prefetch\MAINEMULATORS.EXE-...	SUCCESS	Allc
21:59:...	MainEmulators....	5536	ReadFile	C:\Windows\Prefetch\MAINEMULATORS.EXE-...	SUCCESS	Off
21:59:...	MainEmulators....	5536	ReadFile	C:\Windows\Prefetch\MAINEMULATORS.EXE-...	SUCCESS	Off
21:59:...	MainEmulators....	5536	ReadFile	C:\Windows\Prefetch\MAINEMULATORS.EXE-...	SUCCESS	Off
21:59:...	MainEmulators....	5536	CloseFile	C:\Windows\Prefetch\MAINEMULATORS.EXE-...	SUCCESS	
21:59:...	MainEmulators....	5536	CreateFile	C:	SUCCESS	De
21:59:...	MainEmulators....	5536	QueryInformatio...	C:	BUFFER OVERFL...	Vol
21:59:...	MainEmulators....	5536	CloseFile	C:	SUCCESS	
21:59:...	MainEmulators....	5536	CreateFile	C:\docs\src\emulators	SUCCESS	De
21:59:...	MainEmulators....	5536	CreateFile	C:\Windows\System32\mscoree.dll	SUCCESS	De
21:59:...	MainEmulators....	5536	QueryBasicInfor...	C:\Windows\System32\mscoree.dll	SUCCESS	Cre
21:59:...	MainEmulators....	5536	CloseFile	C:\Windows\System32\mscoree.dll	SUCCESS	
21:59:...	MainEmulators....	5536	CreateFile	C:\Windows\System32\mscoree.dll	SUCCESS	De
21:59:...	MainEmulators....	5536	CreateFileMapp...	C:\Windows\System32\mscoree.dll	FILE LOCKED WI...	Syn
21:59:...	MainEmulators....	5536	CreateFileMapp...	C:\Windows\System32\mscoree.dll	SUCCESS	Syn
21:59:...	MainEmulators....	5536	Load Image	C:\Windows\System32\mscoree.dll	SUCCESS	Ima

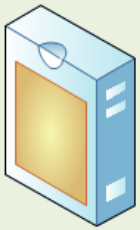
Showing 9,387 of 512,102 events (1.8%) Backed by virtual memory

Registry Audit

File Usage Log

Author: Prof Bill Buchanan

Validating files



```
c:\docs\src\emulators\txt>md5sum *
968c7fc69300e16bb78d3f7082245293 *asa_new.txt
2369d7724a85fc391964ead2ba9dff90 *pixcomplete.txt
d149f53fe45df84f25e58d33a67d393b *pixcomplete2.txt
ac3f9d2ed849933fef1497e032fd8037 *routercomplete.txt
d6fed9075a218e7fd2b30a5fa4003ea4 *routercomplete2.txt
29a97f75a1f09edd128244c4756b18cb *switchcomplete.txt
6fc4116d8b479207ab2afbc6c25a7388 *switchcomplete2.txt
83c978fea02c85f5ecfd840a82a2580c *wireless.txt
302f9ed650903b1a7dec683832b58a9f *wirelesscomplete.txt
0cd328e46813878188d1a35ffcb20f3f *wirelesscomplete2.txt
4f2d0d9e6ad8653f75b30d4ab3c54b26 *wirelesscontrollernew.txt

c:\docs\src\emulators\txt>md5sum *
968c7fc69300e16bb78d3f7082245293 *asa_new.txt
2369d7724a85fc391964ead2ba9dff90 *pixcomplete.txt
d149f53fe45df84f25e58d33a67d393b *pixcomplete2.txt
ac3f9d2ed849933fef1497e032fd8037 *routercomplete.txt
d6fed9075a218e7fd2b30a5fa4003ea4 *routercomplete2.txt
29a97f75a1f09edd128244c4756b18cb *switchcomplete.txt
6fc4116d8b479207ab2afbc6c25a7388 *switchcomplete2.txt
83c978fea02c85f5ecfd840a82a2580c *wireless.txt
46d08a484be99d87a483d7332a8d623a *wirelesscomplete.txt
0cd328e46813878188d1a35ffcb20f3f *wirelesscomplete2.txt
4f2d0d9e6ad8653f75b30d4ab3c54b26 *wirelesscontrollernew.txt
```


Author: Prof Bill Buchanan

Code Forensics

Network-based Behavioural Analysis

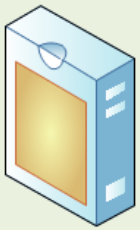
Author: Prof Bill Buchanan

Code Forensics

Network-based Analysis

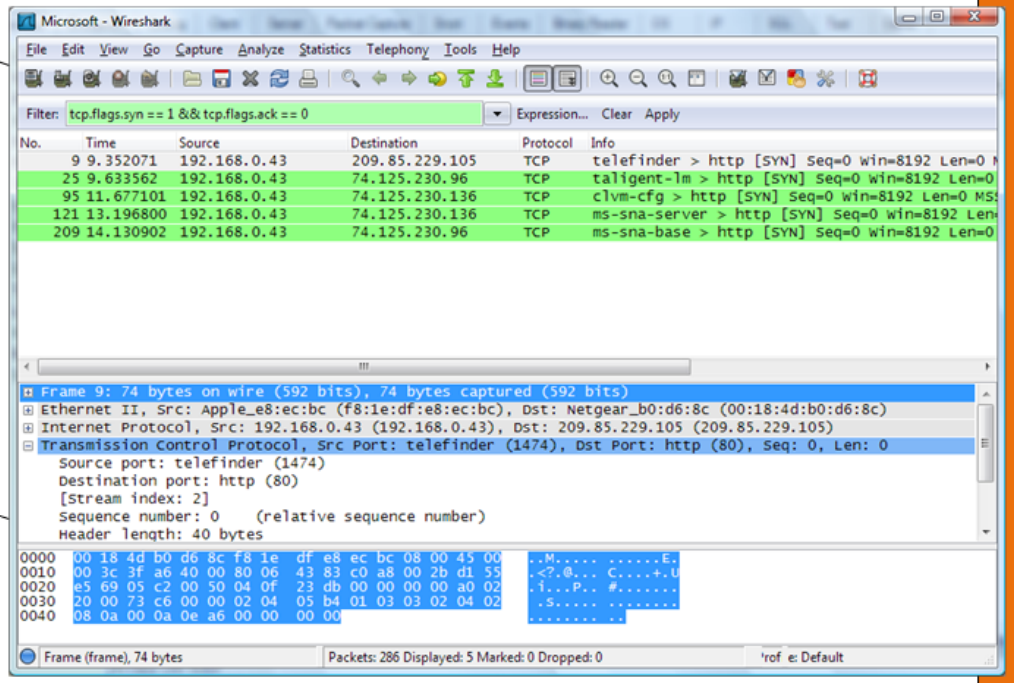
Ports opened

Proto	Local Address	Foreign Address	State
TCP	0.0.0.0:135	Bill-PC:0	LISTENING
TCP	0.0.0.0:445	Bill-PC:0	LISTENING
TCP	0.0.0.0:912	Bill-PC:0	LISTENING
TCP	0.0.0.0:990	Bill-PC:0	LISTENING
TCP	0.0.0.0:1025	Bill-PC:0	LISTENING
TCP	0.0.0.0:1026	Bill-PC:0	LISTENING
TCP	0.0.0.0:1027	Bill-PC:0	LISTENING
TCP	0.0.0.0:1028	Bill-PC:0	LISTENING
TCP	0.0.0.0:1029	Bill-PC:0	LISTENING
TCP	0.0.0.0:1034	Bill-PC:0	LISTENING
TCP	0.0.0.0:5357	Bill-PC:0	LISTENING
TCP	0.0.0.0:17500	Bill-PC:0	LISTENING
TCP	127.0.0.1:1031	Bill-PC:27015	ESTABLISHED
TCP	127.0.0.1:1046	Bill-PC:19872	ESTABLISHED
TCP	127.0.0.1:5354	Bill-PC:0	LISTENING
TCP	127.0.0.1:5679	Bill-PC:0	LISTENING
TCP	127.0.0.1:7438	Bill-PC:0	LISTENING
TCP	127.0.0.1:9089	Bill-PC:0	LISTENING
TCP	127.0.0.1:11880	Bill-PC:0	LISTENING



Network Protocols

Network connections made



Author: Prof Bill Buchanan