APIフックとsysenterフックを利用した新しい解析テクニックとキャッシュカードアクセスのキャプチャリング

NetAgent Co., Ltd.
http://www.netagent.co.jp

Kenji Aiko

API（関数）フックとは

- アプリケーションが、外部ライブラリ（.dll、.so ファイル）に存在する関数（命令コード）を呼び出す際、jmp命令やcall命令などを一時的に変更することで、別の関数へ処理を飛ばすテクニック

- Linux（UNIX系）、Windowsそれぞれで、API フックを実現するライブラリが存在する

APIフックの種類

- 関数の先頭アドレス書き換えによるフック（Detours法）
- プロセスのIAT（Import Address Table）書き換えによるフック
- DLLの置き換えによるフック
- SSDT変更によるSystemServicesフック

他にも多くの方法がある…

Windows CryptoAPI（1/2）

- ADVAPI32.dllが提供する暗号処理系API
- Windows2000以降で利用可能
- 多くの暗号アルゴリズムを、専門知識なしに利用できる
- ハッシュ、署名、検証など、暗号化に関連するライブラリも提供
- Windows環境におけるSSL通信は、内部的にCryptoAPIを利用しているものが多い

Windows Crypt APIs（2/2）

- 暗号系はCrypt***という関数名でエクスポートされている

ADVAPI32.dll がエクスポートしているCrypto系関数群

77D97F96	.text	Export	CryptAcquireContextA
77D985F1	.text	Export	CryptAcquireContextW
77DC0CDA	.text	Export	CryptContextAddRef
77D9A2F9	.text	Export	CryptCreateHash
77D9A7B1	.text	Export	CryptDecrypt
77D9A685	.text	Export	CryptDeriveKey
.....			
77DC1C49	.text	Export	CryptSignHashA
77DC1C39	.text	Export	CryptSignHashW
77D9AB80	.text	Export	CryptVerifySignatureA
77D9B462	.text	Export	CryptVerifySignatureW

デモンストレーション（1/4）

- SSL通信により送受信されるデータがCryptoAPIを通るため、プロセスのCrypto系関数をフックすれば、SSL通信をキャプチャできる

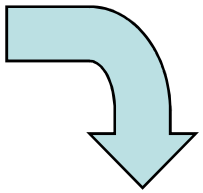
Demo 1

IEのSSL通信をキャプチャする

SSLにより暗号化されたデータ

IEのSSL（https）通信をWiresharkでキャプチャリングすると以下のように見える

Protocol	Info
SSLv3	Client Hello
SSLv3	Server Hello, Certi
SSLv3	Client Key Exchange
SSLv3	Change Cipher Spec,
SSLv3	Application Data
SSLv3	Application Data



detail

http通信が暗号化されている！

```
Secure Socket Layer
  SSLv3 Record Layer: Application Data Protocol: http
      Content Type: Application Data (23)
      Version: SSL 3.0 (0x0300)
      Length: 331
      Encrypted Application Data: 35204A95F1183D47C673ACAF929FBECD68E0844055911D3D...
```

Encrypted Data

CryptoAPIを通ったデータ

- Crypt Encrypt/Decrypt を通ったデータが平文として確認できる

■send data

```
-- CryptEncrypt --↓
GET / HTTP/1.1↓
Accept: image/gif, image/x-xbitmap,
wave-flash, application/vnd.ms-powe
/msword, */*↓
Accept-Language: ja,en-us;q=0.5↓
Accept-Encoding: gzip, deflate↓
User-Agent: Mozilla/4.0 (compatible
.4322; .NET CLR 2.0.50727)↓
Host: www.netagent.co.jp↓
Connection: Keep-Alive↓
↓
筌/ZA~ X苳・ｹ朮uI↓
```

■recv data

```
-- CryptDecrypt --↓
HTTP/1.0 200 OK↓
Date: Wed, 08 Oct 2008 13:52:05 GMT↓
Server: Apache/1.3.33 (Debian GNU/Li
Last-Modified: Mon, 20 Jun 2005 03:0
ETag: "3941-13-42b6325d"↓
Accept-Ranges: bytes↓
Content-Length: 19↓
Connection: close↓
Content-Type: text/html↓
↓
ssl.netagent.co.jp↓
鈁/!鈎<VヒモCⅥ・ ↓
```

SSL通信の安全性

- 暗号化されているため、通信を傍受しただけでは内容を確認できない

- MITM（Man In The Middle）による通信の傍受は可能だが、正当なセキュリティ証明書を使って通信を行うことで、通信の信頼性、安全性は保たれる

APIフックの汎用性

- アプリケーションがSSL通信を復号する瞬間をフックすれば、SSLの内容を確認できる
- SSLの通信内容を改変することも可能
- 内容を改変する場合も、セキュリティ証明書の正当性は関係ない

APIフックはとても簡単

- Linux（UNIX系）OSでは、LD_PRELOAD環境変数を利用することで、実現できる

- Windowsでは、Microsoft Research Teamからリリースされているdetoursライブラリをインストールすることで実現できる

Detoursライブラリ

http://research.microsoft.com/sn/detours/

LD_PRELOAD

- Linux(UNIX)環境で使用可能
- 使用方法は、LD_PRELOAD環境変数に、対象となるsoファイルを登録するだけ

```
-----  terminal
% gcc –shared –fPIC –o intercept.so intercept.c –ldl
% LD_PRELOAD=./intercept.so target_prog
-----
```

Detoursライブラリ（1/3）

- Microsoft Research Teamから提供されているライブラリ
- ターゲット関数の先頭数バイトを書き換えることでAPIフックを行う
- 仕組みはシンプルで使いやすい

Detoursライブラリ（2/3）

- CryptEncrypt関数をフックする例

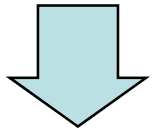
  ADVAPI32.dll内にあるCryptEncrypt関数の先頭バイトを以下に示す

```
77DA1558    6A 24           PUSH 24
77DA155A    68 1016DA77     PUSH ADVAPI32.77DA1610
77DA155F    E8 B553FEFF     CALL ADVAPI32.77D86919
77DA1564    33FF            XOR EDI,EDI
```

- これは一般的な命令コードだが、Detoursによって関数フックを行うと、命令コードが次のように書き換わる

Detoursライブラリ（3/3）

```
77DA1558   6A 24            PUSH 24
77DA155A   68 1016DA77      PUSH ADVAPI32.77DA1610
77DA155F   E8 B553FEFF      CALL ADVAPI32.77D86919
77DA1564   33FF             XOR EDI,EDI
```



detours.dllにより、先頭数バイトが変更される

```
77DA1558 - E9 33062698      JMP CryptCap.?Mine_CryptEncrypt
77DA155D   CC               INT3
77DA155E   CC               INT3
77DA155F   E8 B553FEFF      CALL ADVAPI32.77D86919
77DA1564   33FF             XOR EDI,EDI
```

- detours.dllによって、CryptEncrypt関数の先頭5バイトがjmp命令に変更された

IAT書き換えによる関数フック

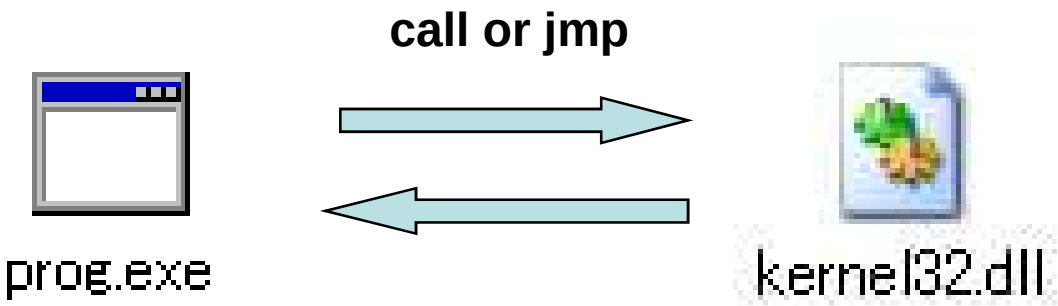
- プロセス内のIAT（Import Address Table）を変更し、別の関数へ飛ばす方法

（※）「Advanced Windows」 by Jeffrey Richter
に詳細が記述されている。

DLLフック（1/2）

- ターゲットとなる関数を持つDLLと同じエクスポート関数を持つ偽のDLLを作成し、偽のDLLをロードさせるようにEXEファイルを変更することでAPIフックを行う方法（DLLフック）

■ **normal**



DLLフック（2/2）

- prog.exeとkernel32.dllの間にダミーとなる関数を用意することで、APIフックを行える

SystemServicesフック（1/2）

- SSDT(System Service Descriptor Table)を変更することによるSystemService（システムコールと同義）のフック

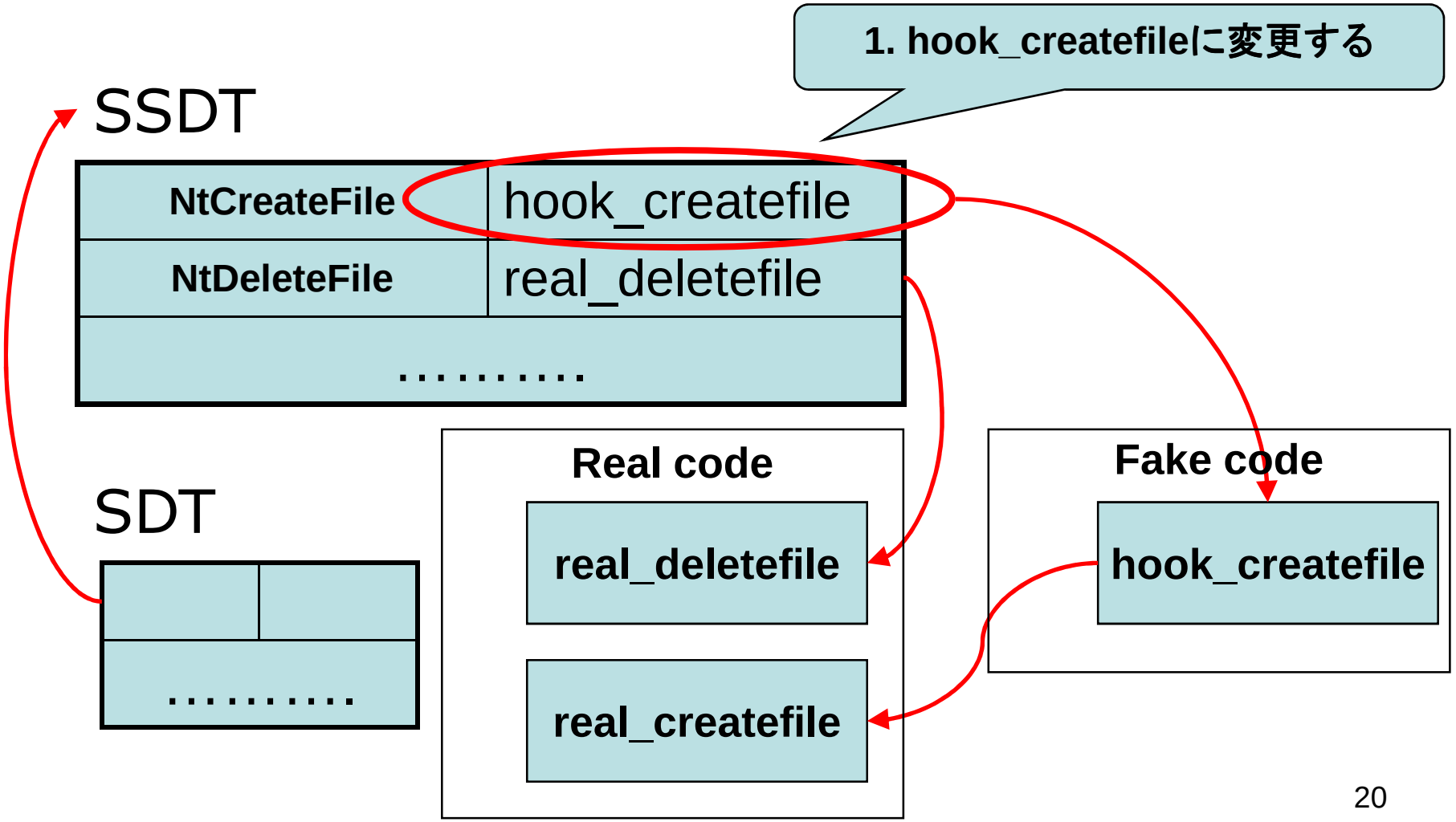
- 処理はカーネルランドで行う

Hooking Windows NT System Services

http://www.windowsitlibrary.com/Content/356/06/2.html

に詳細が記述されている。

SystemServicesフック（2/2）



sysenterフック（1/4）

- WindowsXP/2003(x86)環境では、sysenter命令によりカーネルへ処理が渡る
- sysenterはntdll.dll内で呼び出される
- sysenterはMSRに設定された値にジャンプする

Sysenterフック（2/4）

ntdll.dll（ZwCreateFile）

```
7C94D682   B8 25000000      MOV EAX,25
7C94D687   BA 0003FE7F      MOV EDX,7FFE0300
7C94D68C   FF12             CALL NEAR DWORD PTR DS:[EDX]
7C94D68E   C2 2C00          RETN 2C
```

Jump!

ntdll.dll（sysenter）

```
7C94EB8B   8BD4             MOV EDX,ESP
7C94EB8D   0F34             SYSENTER
```

eaxレジスタの値がシステムコール番号

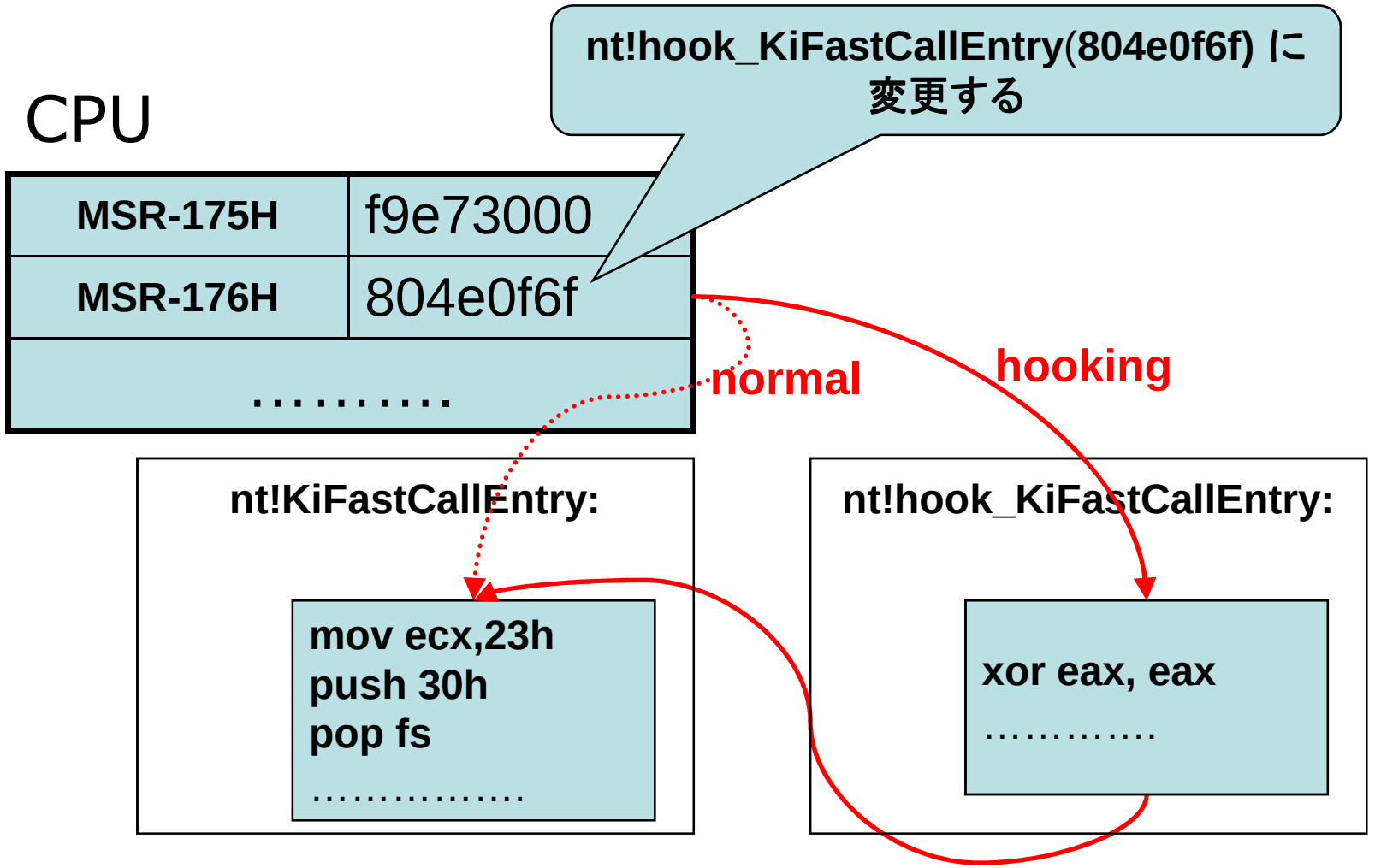
Sysenterフック（3/4）

■sysenter実行時の処理

1. CSレジスタに (MSR-174H) の値 をロード
2. EIPレジスタに (MSR-176H) の値 をロード
3. SSレジスタに (MSR-174H) の値+8 をロード
4. ESPレジスタに (MSR-175H) の値 をロード

つまり、CPUごとの(MSR-176H)を変更することでsysenterフックが可能

sysenterフック（4/4）



デモンストレーション（2/4）

- APIフックを利用することで、MITM（Man In The Middle）による通信の傍受も可能

Demo 2

P2Pソフトの通信をキャプチャする

電子マネー Edy（エディ）

- プリペイド型の電子マネー（suicaと同じ）
- 銀行口座からEdyへ入金（チャージ）できる
- Edyの残高を専用のソフトによって確認、再入金できる

FeliCaポート（PaSoRi）

- SONYが開発した、ICカードを直接パソコンで読み込むシステム
- USBで外付けできるものも販売されている
- FeliCaポートを利用して、ICカードの情報を読み込むライブラリもある（BSDライセンス）

EdyViewer.exe

- Edyの情報を確認、管理するソフトウェア
- ユーザーの銀行口座を登録することで、口座からEdyへの入金を行える
- Windows上で動作する
- 公式のFeliCaポート対応ソフトウェア

felicalibライブラリ

- USB接続の読み込みツール（PaSoRi）を使い、ICカードへアクセスするためのライブラリ（BSDライセンスで提供）

  http://felicalib.tmurakam.org/

- Suica、Edy、nanacoなどの電子マネーへのアクセスに使用できる。

- 非公式のFeliCaポート対応ライブラリ

デモンストレーション（3/4）

- felicalibを利用することで、ICカードの情報を閲覧するツールを作成できる

Demo 3

ICカードの情報を取得する

ICカードのセキュリティ（1/2）

- 読み込み可能なブロックと不可のブロックがある
- 一部、暗号化されているブロックがある
- felicalibでは、書き込みができない
- felicalibでは、暗号化ブロックへアクセスできない

ICカードのセキュリティ（2/2）

- 公式ツールEdyViewer.exeでは、暗号化部分の読み込み、書き込みを含め、すべてが可能

- 公式ツールEdyViewer.exeは、管理サーバへの通信にSSL（https）を使用

デモンストレーション（4/4）

- ICカードへの入金（チャージ）時に発生するSSL通信を見る

Demo 4

公式ツールのSSL通信をキャプチャ

今後の展望（1/2）

- APIフックならば、ユーザーランドとカーネルランドの通信もキャプチャできる

- EdyViewer.exeとFeliCaポートドライバとの会話をキャプチャできないか？

今後の展望(2/2)

- sysenterフックで、システムコール呼び出しを監視できる
- システムコール呼び出し履歴から、通常のAPIの呼び出し履歴を推測できないか？

Thank you!

Any questions?