Monthly Research

Windowsのセキュリティ新機能 Control Flow Guardについて

株式会社ＦＦＲＩ
http://www.ffri.jp

Ver 1.00.01

Control flow guard(Guard CF)の概要

- Windows 8.1 Previewで試験的に導入された、新しいセキュリティ機能
  - Windows 8.1 RTM(Release To Manufacturing)ではOFFとなり、現在リリースされている8.1では利用不可
  - Windows 10 Technical Preview, Windows 8.1 Update（拡張パック）にて利用可能
- 以降、Control flow graph(CFG)と混同させないため、 Guard CFと記述

注意事項

- Control flow guardは開発・検証中の技術であり、Windows10のリリース版に実装されるかどうかは不明

- 本資料では、Windows 10 Technical Preview とVisual Studio 2015 Previewを用いて検証をおこなった

脅威モデル

- 脆弱性を利用した制御フローの奪取を防ぐため、信頼できないアドレスに対して関数呼び出しを止める
  - indirect call (call eaxや call [EBP+var8]など）が保護対象

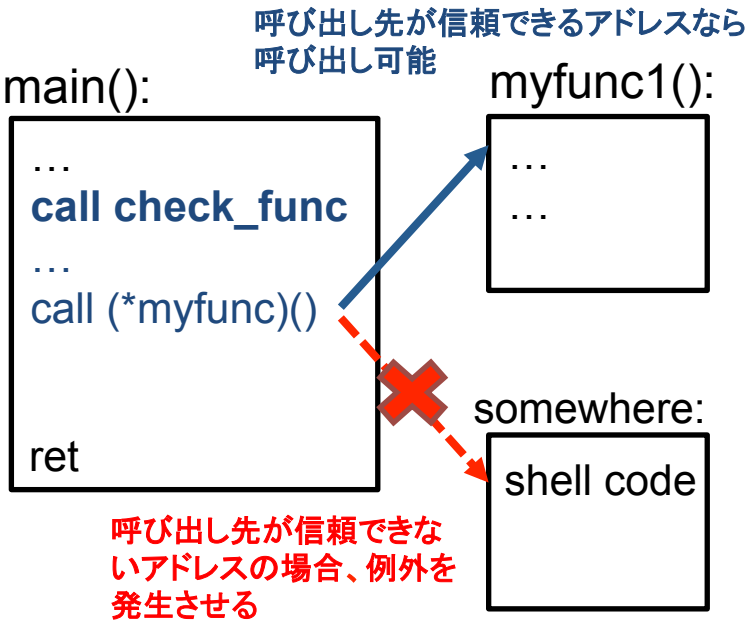
- 典型的な攻撃例
  - vtable overwrite

Guard CFによる保護(1)

- 間接関数呼び出し時に、呼び出し先アドレスが信頼できる関数かどうかを検査する
  - 信頼できないアドレスの場合、例外を発生させる

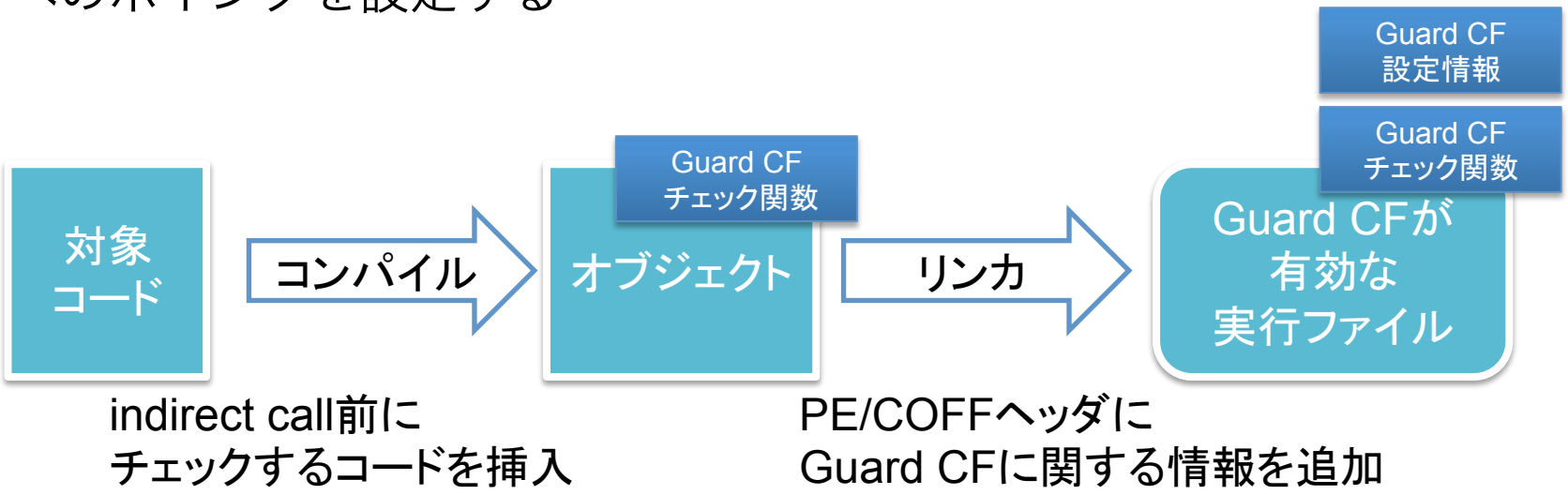
```
void myfunc1() {
    printf("myfunc1\n");
}

int main(int argc, char* argv[])
{
    void(*myfunc)();
    myfunc = myfunc1;
    (*myfunc)();
    return 0;
}
```

コンパイル時にコード挿入
リンカー保護情報を埋め込む



Guard CFによる保護(2)

- 呼び出し先アドレスがGuard CF Function Tableに登録されている場合、信頼できるものとする
- リンカーがPE/COFFヘッダにGuard CF Function Tableやloadconfig情報を格納する
- プログラムのロード時にntdll内の関数でbitmapの作成、チェック関数へのポインタを設定する



In Visual Studio 2015 Preview

- コンパイラオプションで以下のオプションを指定
  - 隠しオプションとなっている

```
cl /d2guard4  test.cpp  /link /guard:cf
```

参考：http://blogs.msdn.com/b/vcblog/archive/2014/12/08/visual-studio-2015-preview-work-in-progress-security-feature.aspx

Guard CFに関するPE/COFFヘッダ(1)

- DLL Characteristics

```
OPTIONAL HEADER VALUES
        10B magic # (PE32)
            …
      C140 DLL characteristics
            Dynamic base
            NX compatible
            Guard
            Terminal Server Aware
```

Guard CFを有効にした場合

```
OPTIONAL HEADER VALUES
        10B magic # (PE32)
            …
      8140 DLL characteristics
            Dynamic base
            NX compatible
            Terminal Server Aware
```

Guard CFを無効にした場合

Guard CFに関するPE/COFFヘッダ(2)

- Load config structure

```
Section contains the following load config:

            0000005C size
            ...
            0041D108 Guard CF address of check-function pointer
            00000000 Reserved
            0041D150 Guard CF function table
                  2A Guard CF function count
            00003500 Guard Flags
                     CF Instrumented
                     FID table present
                     Protect delayload IAT
                     Delayload IAT in its own section
```

Guard CFを有効にした場合、上記のような情報が追加される

Guard CFに関するPE/COFFヘッダ(3)

- Load config structure

```
…
Guard CF Function Table

        Address
        --------
        00401000
        00401030
        004011E0
        00401270
        004013F0
        …
```

- リンカがindirect callされる関数をリストアップし、 Guard CF function tableを作成する

Guard CFの動作確認(1)

- サンプルプログラムとGuard CFチェック関数の挿入の確認

```
int main(int argc, char* argv[])
{
        void(*myfunc)();
        myfunc = myfunc1;
        (*myfunc)();
        return 0;
}
```

サンプルコード

```
.text:00401050        push    ebp
.text:00401051        mov     ebp, esp
.text:00401053        sub     esp, 8
.text:00401056        mov     [ebp+var_8], offset sub_401030
.text:0040105D        mov     eax, [ebp+var_8]
.text:00401060        mov     [ebp+var_4], eax
.text:00401063        mov     ecx, [ebp+var_4]
.text:00401066        call    j__guard_check_icall_fptr
.text:0040106B        call    [ebp+var_4]
.text:0040106E        xor     eax, eax
.text:00401070        mov     esp, ebp
.text:00401072        pop     ebp
.text:00401073        retn
```

IDAでの逆アセンブル結果

追加されるGuard CFチェック関数

Guard CFの動作確認(2)

- チェック関数の動作
  - bitmapを参照し、関数が登録されていたらチェックをパス
  - 信頼出来ないアドレスだった場合
    - Security assertion（int 29h）例外を発生させる

Address	Type	Size	Committed	Private	Total WS	Private ...	Sharea...	Share...	Lock...	Blocks	Protection	Details
001F0000	Shareable	64 K	64 K		4 K		4 K			1	Read/Write	
00220000	Shareable	76 K	76 K		72 K		72 K	72 K		1	Read	
00240000	Thread Stack	256 K	44 K	44 K	12 K	12 K				3	Read/Write/Guard	64-bit thread stack
00280000	Thread Stack	1,024 K	20 K	20 K	12 K	12 K				3	Read/Write/Guard	Thread ID: 5068
00380000	Shareable	16 K	16 K		16 K		16 K	16 K		1	Read	
00390000	Private Data	8 K	8 K	8 K	8 K	8 K				1	Read/Write	
003A0000	Mapped File	728 K	728 K		128 K		128 K	128 K		1	Read	C:\Windows\System32\locale.nls
00520000	Private Data	64 K	20 K	20 K	20 K	20 K				2	Read/Write	
00610000	Heap (Private Data)	1,024 K	48 K	48 K	48 K	48 K				2	Read/Write	Heap ID: 1 [COMPATABILITY]
00AB0000	Image (ASLR)	168 K	168 K	28 K	112 K	16 K	96 K			5	Execute/Read	C:\Users\Yosuke\Desktop\cfgtest\bin\cfgtest.exe
00AE0000	Shareable	32,768 K	6,176 K		44 K	20 K	24 K	4 K		12	Read	
00AE0000	Shareable	56 K									Reserved	
00AEE000	Shareable	28 K	28 K		12 K	12 K					Read	
00AF5000	Shareable	84 K									Reserved	
00B0A000	Shareable	8 K	8 K		8 K	8 K					Read	
00B0C000	Shareable	24,300 K									Reserved	
022C7000	Shareable	5,580 K	5,580 K								No access	
0283A000	Shareable	24 K	24 K		8 K		8 K				Read	
02840000	Shareable	348 K	348 K								No access	
02897000	Shareable	16 K	16 K		4 K		4 K				Read	
0289B000	Shareable	128 K	128 K								No access	
028BB000	Shareable	44 K	44 K		12 K		12 K	4 K			Read	
028C6000	Shareable	2,152 K									Reserved	
75650000	Image (ASLR)	1,404 K	1,404 K	28 K	236 K	16 K	220 K	220 K		4	Execute/Read	C:\Windows\SysWOW64\KernelBase.dll
76DC0000	Image (ASLR)	896 K	576 K	16 K	156 K	12 K	144 K	144 K		12	Execute/Read	C:\Windows\SysWOW64\kernel32.dll

確保された
bitmap領域

vmmapによるプロセスのメモリ使用状況の確認

Guard CFの防御範囲

- 現状はcall命令のみが対象
  - 間接ジャンプ命令やreturn命令によって制御が奪われるような攻撃が成立してしまう
  - また、間接呼び出しされる関数はどのcallからでも呼び出せるため、Code-reuse attackには限定的にしか対応できない

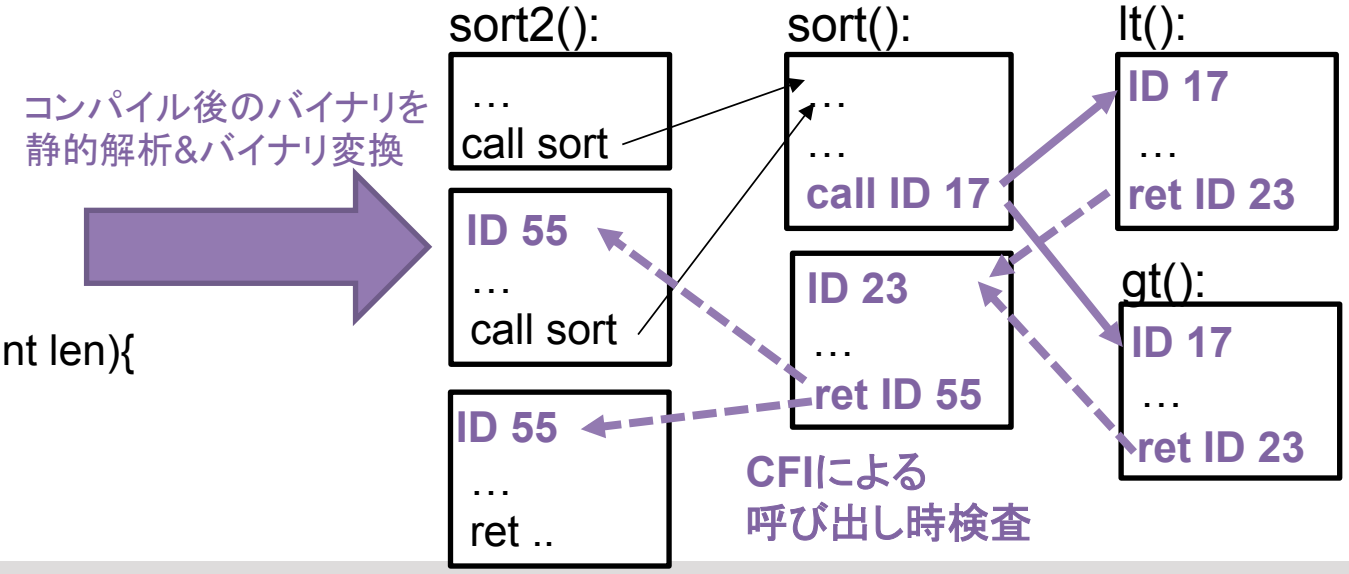
- ただし、Microsoftが提供する脆弱性緩和ツールであるEMETにROP攻撃緩和機能があるため、併用することを想定していると考えられる

参考：Control Flow Integrity（CFI）

- 間接分岐、間接呼び出しの飛び先とリターンアドレスを検証するコードをプログラムに挿入する
  - Martín Abadi とMSの研究者が2005年に発表した技術

- プログラムを静的解析し、実行前に決定できるcompute jumpやindirect call、returnの前に、対応するIDとID検証コードを挿入することで実現

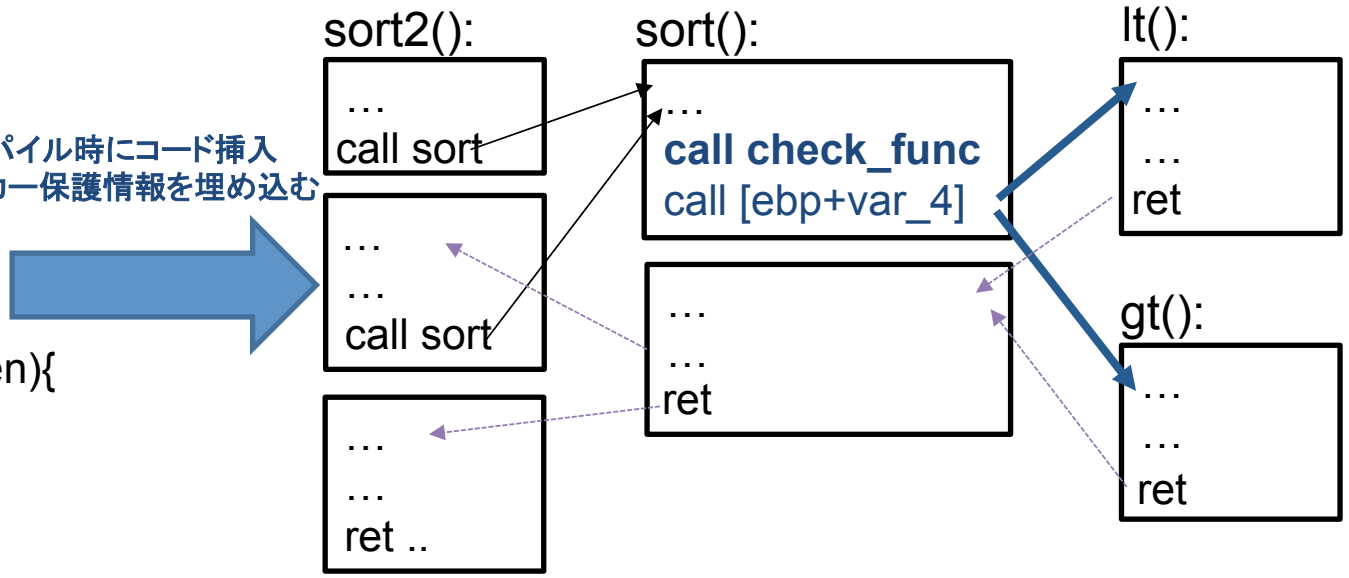
```
bool lt(int x, int y) {
  return x < y;
}
bool gt(int x, int y) {
  return x > y;
}
sort2(int a[ ], int b[ ], int len){
  sort( a, len, lt );
  sort( b, len, gt );
}
```



CFIとGuard CFの関係

- CFIはGuard CFよりも強い保護が可能
  - しかし、バイナリ変換が必要なことや性能低下、互換性の点で実用が難しかったと推測できる
- Guard CFでは、CFIを簡略化し、間接関数呼び出し時に呼び出しアドレスが信頼できるかどうかのみをチェックしている

```
bool lt(int x, int y) {
  return x < y;
}
bool gt(int x, int y) {
  return x > y;
}
sort2(int a[ ], int b[ ], int len){
  sort( a, len, lt );
  sort( b, len, gt );
}
```



まとめ

- Windowsの新しいセキュリティ機能であるControl flow guard(Guard CF)について紹介した
  - 対応するコンパイラとリンカでプログラムをビルドし、かつ対応しているOS上で実行することで初めて有効となる

- Control flow guardは１０年来の研究成果を自社の商用OSに適用し、取り入れた意欲的なセキュリティ機能であるといえる

参考文献

- “Visual Studio 2015 Preview: Work-in-Progress Security Feature” http://blogs.msdn.com/b/vcblog/archive/2014/12/08/visual-studio-2015-preview-work-in-progress-security-feature.aspx (2014/12/19 viewed)
- MJ0011, "Windows 10 Control Flow Guard Internals“, Power of Community 2014.
- Martín Abadi, Mihai Budiu, Úlfar Erlingsson, and Jay Ligatti, “Control-Flow Integrity”, ACM CCS’05, November 2005 http://research.microsoft.com/apps/pubs/default.aspx?id=64250

Contact Information

E-Mail : research—feedback@ffri.jp
Twitter : @FFRI_Research