10. Controlled-Channel Attacks

Ao Sakurai

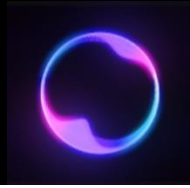
2026年度セキュリティキャンプ全国大会
L1 – 暗号・CVMビルド&スクラップゼミ

本セクションの目標

- SGXを初めとしたTEEに対する強力な攻撃である、Controlled-Channel攻撃についての解説を行う
- Controlled-Channel攻撃をTDXに拡張し、TDX特有の制約をくぐり抜けながらPoC攻撃を成立させる攻撃の実践を行う
- 同時に、Controlled-Channel攻撃に対する防御策の実装に関しても議論する

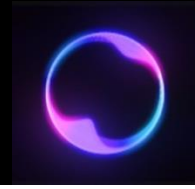
Controlled-Channel攻撃

Controlled-Channel攻撃



- **SGX**のような**防護システム**に対して**非常に有効**な**サイドチャネル攻撃**の一つ
- この攻撃では**ページフォルト**を悪用する
- 以下、OSは**攻撃者**により**制御権を掌握されている**と仮定する
- この攻撃では**3つ**のフェーズが存在する：
  1. コード・実行バイナリに対する**オフライン解析**
  2. **オンライン解析** (①実行バイナリを実行; ②ページフォルト起動; ③観測)
  3. 観測結果から**秘密を推定**

ページング

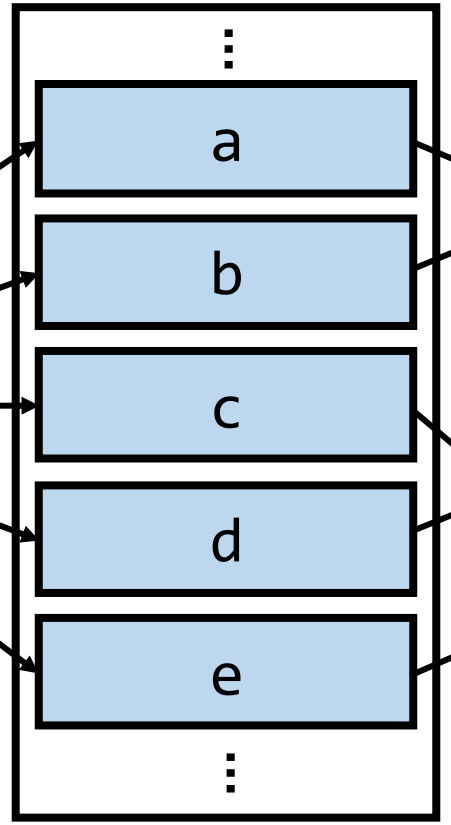


- 仮想記憶を実装するためのアルゴリズム

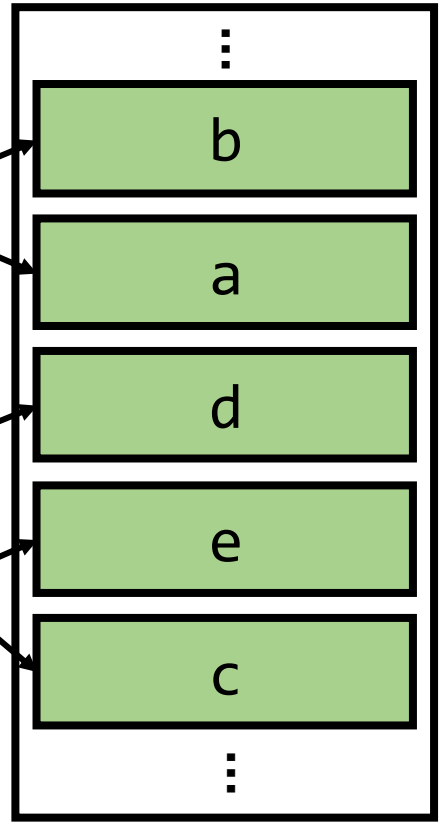
```
int main(void)
{
    uint8_t a[4096];
    uint8_t b[4096];
    uint8_t c[4096];
    uint8_t d[4096];
    uint8_t e[4096];

        ⋮
```

コード

※極端な例

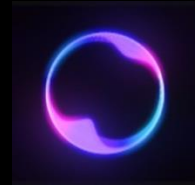


仮想アドレス

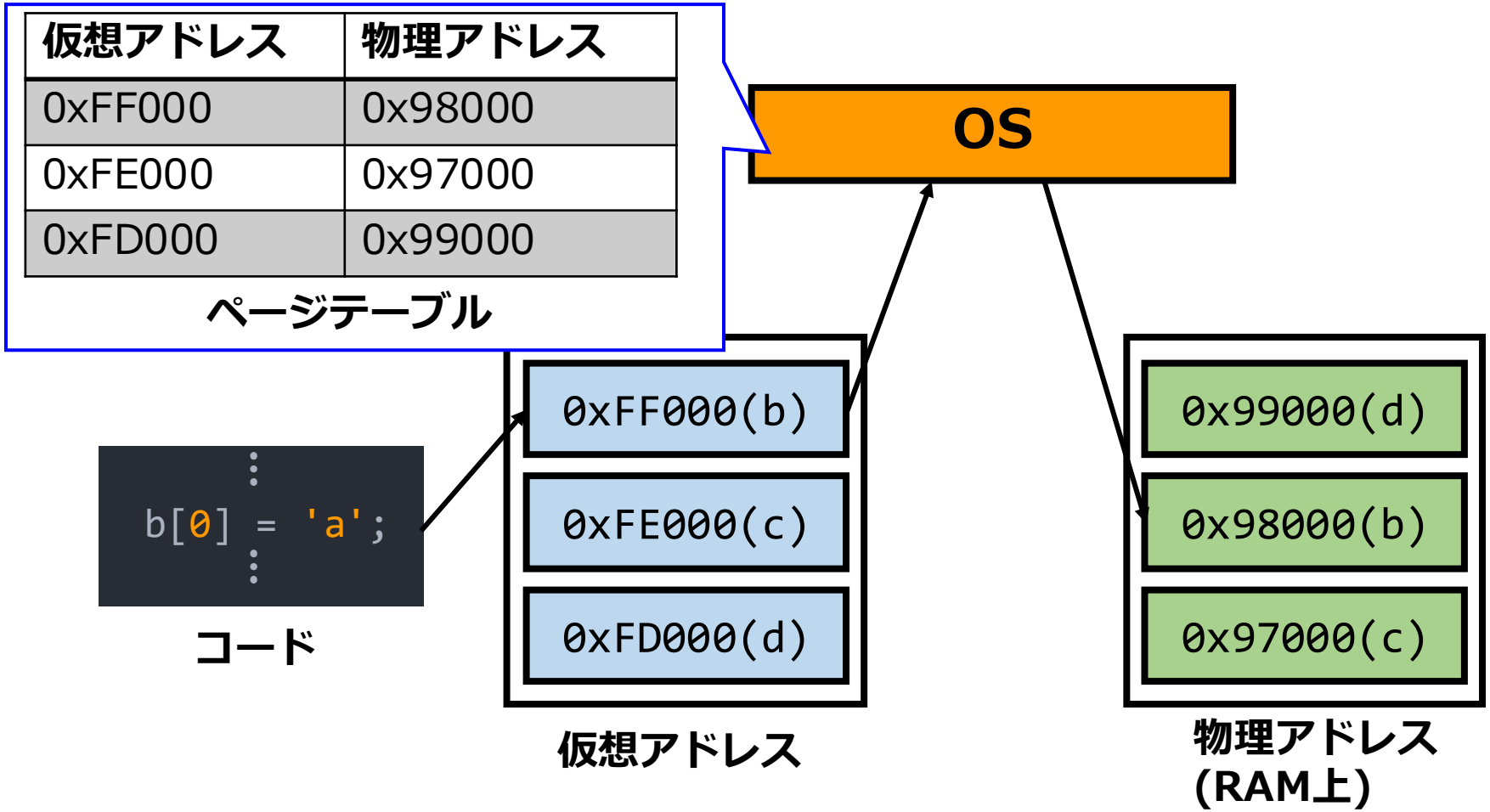


物理アドレス
(RAM上)

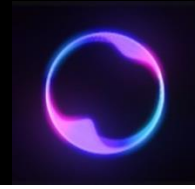
ページテーブル



- OSは**ページテーブル**を用いて**仮想アドレス**に対応する**物理アドレス**を取得する

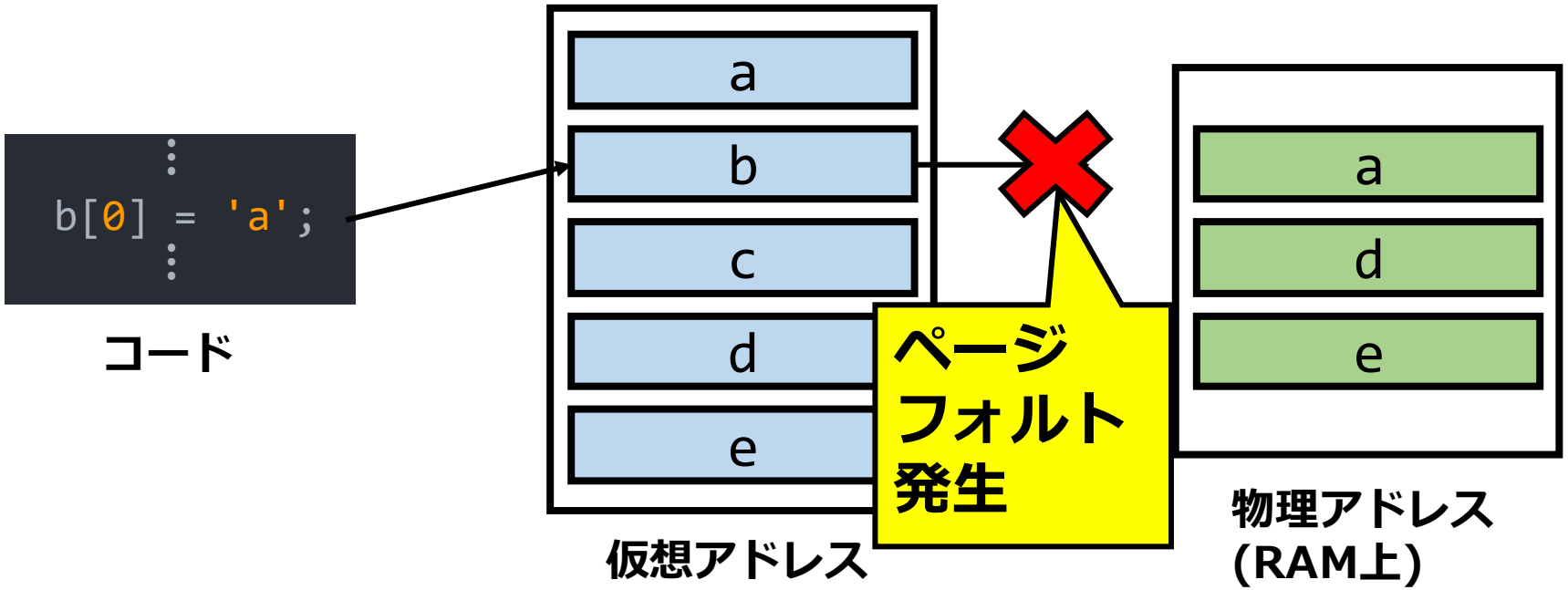


ページフォルト

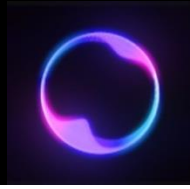


- **ページフォルト**: 以下の状況が発生した場合に発生する、**必ずしも致命的ではない**エラーの事

1. プログラムが**仮想ページ**にアクセスする
2. OSが何らかの原因 (ページアウト、**アクセス制限**等)により**物理アドレス**を**仮想アドレス**から**導出できない**



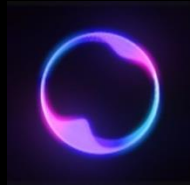
入力依存処理



- ある**処理実行**が**特定の入力データ**によって**条件的に決定される**時、それを**入力依存処理**と呼ぶ
  - 例: ifブロックに囲まれた処理

- 入力依存処理には**2種類**存在する：
  - 入力依存**制御転送**
  - 入力依存**データアクセス**

入力依存制御転送



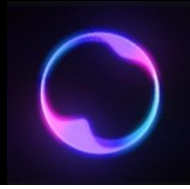
- **変数s**によって**どちらの関数**にアクセスされるかが決定される時、それを「**入力依存制御転送**」と呼ぶ

```
char* WelcomeMessage(GENDER s) {
  char *mesg;

  //GENDER is an enum of MALE and FEMALE
  if(s == MALE) {
    mesg = WelcomeMessageForMale();
  }
  else {
    mesg = WelcomeMessageForFemale();
  }

  return mesg;
}
```

入力依存制御転送

入力依存制御転送

入力依存データアクセス



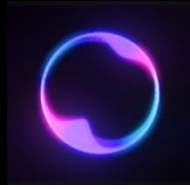
- **変数s**によって**どちらのデータ**にアクセスされるかが決定される時、それを「**入力依存データアクセス**」と呼ぶ

```
void* CountLogin(GENDER s) {

  if(s == MALE) {
    gMaleCount++;
  }
  else {
    gFemaleCount++;
  }

  return;
}
```

入力依存データアクセス

入力依存データアクセス

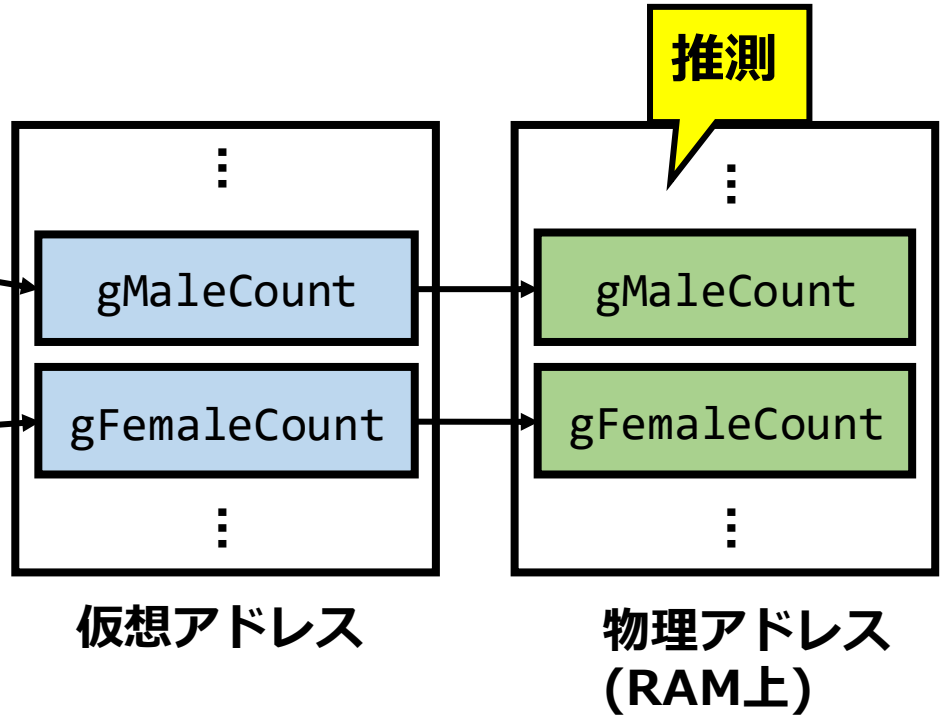
Phase 1. オフライン解析



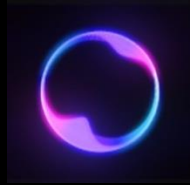
- **実行バイナリ**や**ソースコード**を解析し、 gMaleCountや gFemaleCountがロードされる**アドレスを推測**する

```
void* CountLogin
(GENDER s) {

  if(s == MALE) {
    gMaleCount++;
  }
  else {
    gFemaleCount++;
  }

  return;
}
```

コード

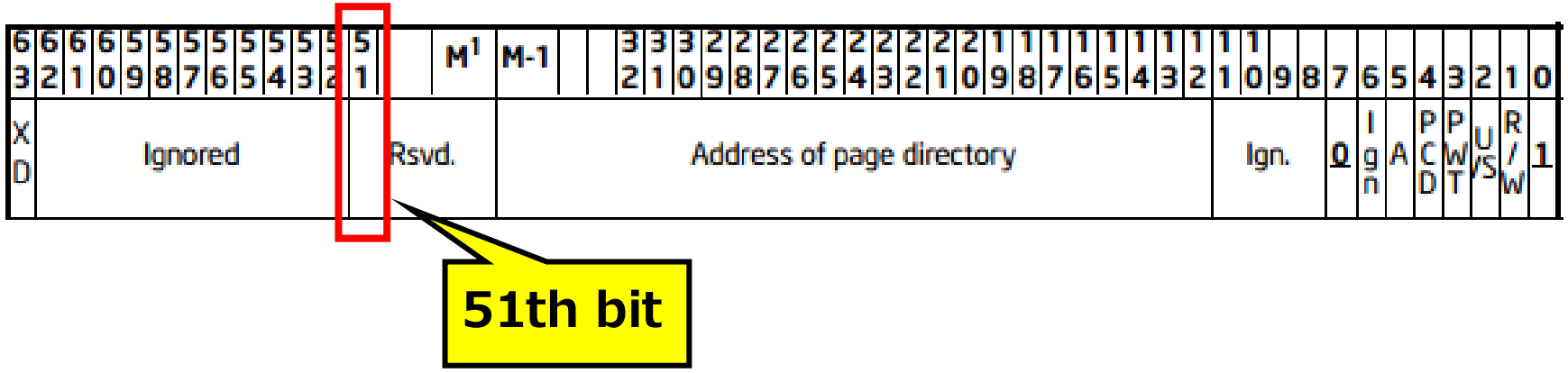


Phase 2. ページフォルト起動 (1/2)

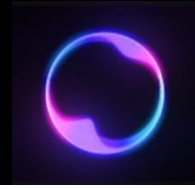


- SGXはページテーブルを**OSと共有**している為、ページテーブルは**容易に改竄できる**

- X86-64のページテーブルエントリの**51ビット目**を立てる事で、**対応するページ**へのアクセスを**全て制限できる**

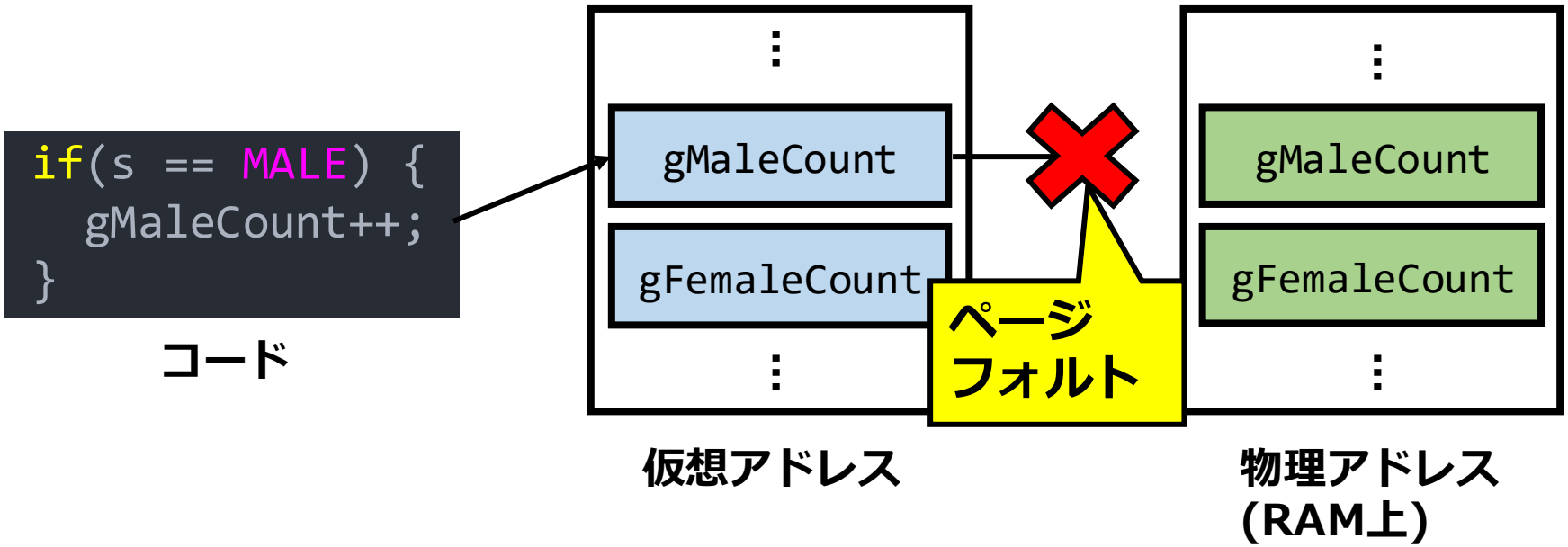


Phase 2. ページフォルト起動 (2/2)

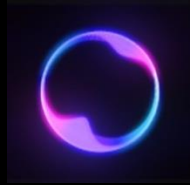


- ページテーブルエントリの**予約ビット**を立てる等により、gMaleCount及びgFemaleCountの載る**ページへのアクセス**を**禁止**する

- **禁止したページ**に載る**データへのアクセス**が発生すると、**ページフォルト**が**発生**する

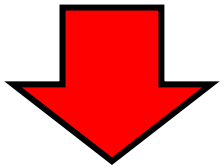


Phase 3. 秘密情報の推定



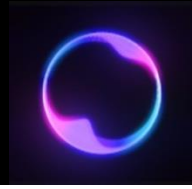
- **ページフォルト**が発生すると、**OS**の**ページフォルトハンドラ**に**ページフォルトアドレス**が伝達される

- 攻撃者は、**攻撃対象の入力依存要素**が載っている**アドレス**(あるいは**ページ**)を**オフライン解析**にて取得済みである



ページフォルトアドレスをOS**から**取得**する事により、**
gMaleCountへのアクセスが発生**した事、ひいては**
“s”がMALE**だった事も漏洩してしまう**

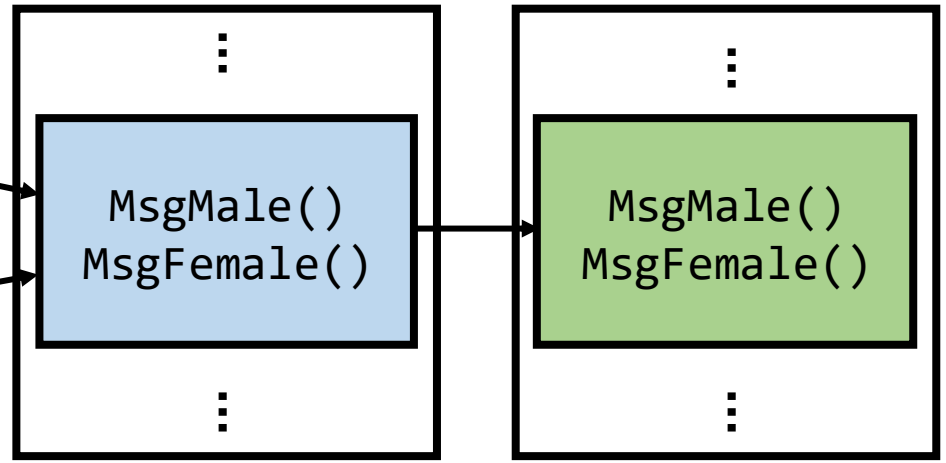
この攻撃を実行する上での困難



- **複数のコードやデータ**が**同一ページ上に載る**可能性が**多分に存在する**ため、その場合単にフォールトの発生したページだけを見るだけではどちらのコードやデータなのか**識別できない**

```
char* WelcomeMessage
 (GENDER s) {
  char *mesg;

  if(s == MALE) {
    mesg = MsgMale();
  }
  else {
    mesg = MsgFemale();
  }

  return mesg;
}
```

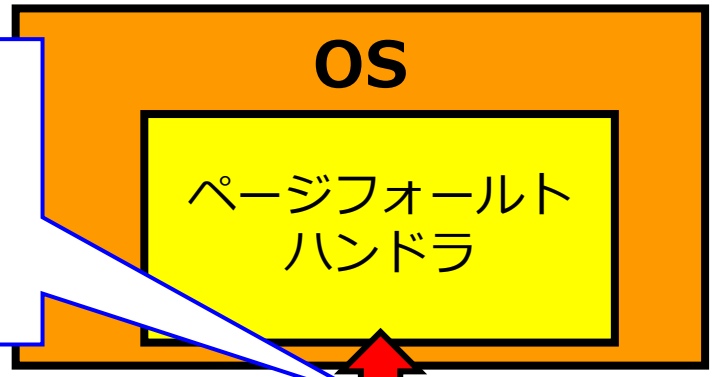
コード



SGXから返されるページフォールトアドレス

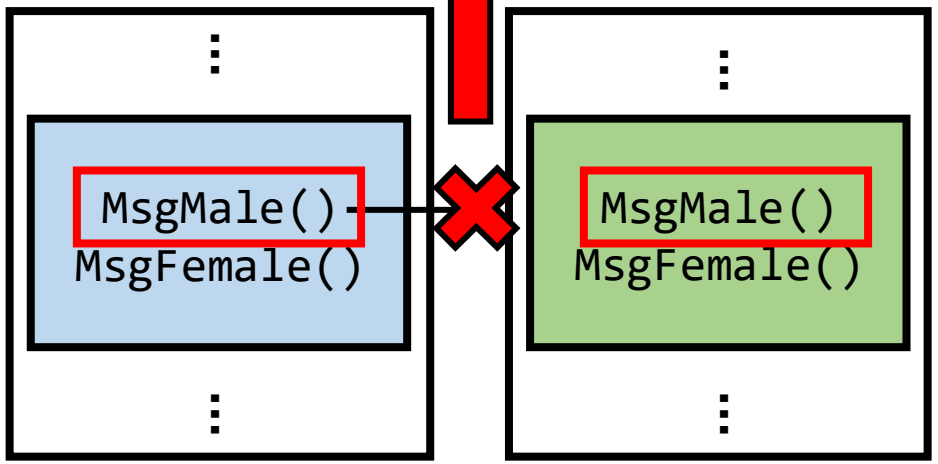
OSが制御可能

通知されるフォールトアドレスは
MsgMale()の完全なアドレスではなく、下位12ビットがゼロ化されている

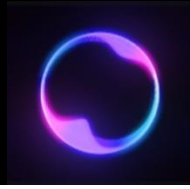


**OSはページフォールトの
発生したページ番号の粒度
でしかアドレスを取得できない**

**OSは制御不能
（Enclave）**



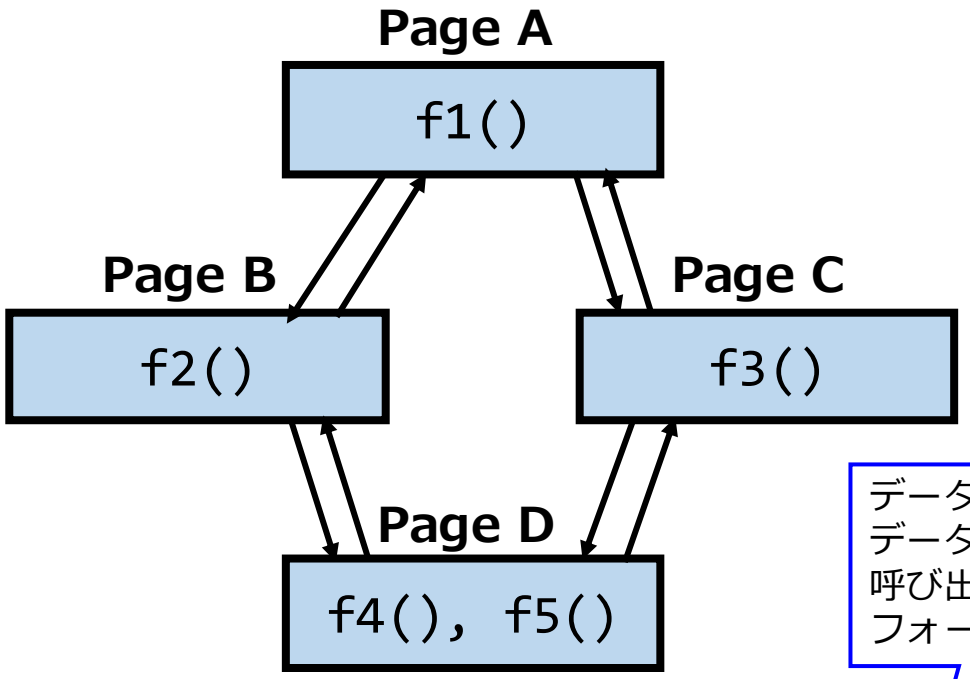
この攻撃を実行する上での困難



- **ページレベルの粒度**でしかページフォールトアドレスからは情報を得られないため、**複数の秘密情報**が**同一ページ上**に**載っている**と、このままでは**識別が出来ない**

- この問題を解決するには、次のページで説明する**ページフォールトシーケンス**という概念を用いる

ページフォールトシーケンス



ページレベルで見た場合の制御転送

A, B, D, B, A, C, D, C, A

f4()

f5()

コードページフォールトシーケンス

データを攻撃対象とする場合は、
データアクセスの直前の関数
呼び出しに対応するコードページ
フォールトシーケンスを頼りにする

開始点

```
f1 {
   …
   f2();
   …
   f3();
   …
}
```

```
f2 {
   …
   f4();
   …
}
```

```
f3 {
   …
   f5();
   …
}
```

ソースコード

攻撃実践例 – Hunspell（スペルチェッカ）

- Hunspellにより参照される辞書ハッシュテーブルに対して攻撃し、秘密情報である文章を抽出

```
Folklore, legends, myths and fairy tales
have followed childhood through the ages,
for every healthy youngster has a wholesome
and instinctive love for stories fantastic,
marvelous and manifestly unreal. The winged
fairies of Grimm and Andersen have brought
more happiness to childish hearts than all
other human creations.
```

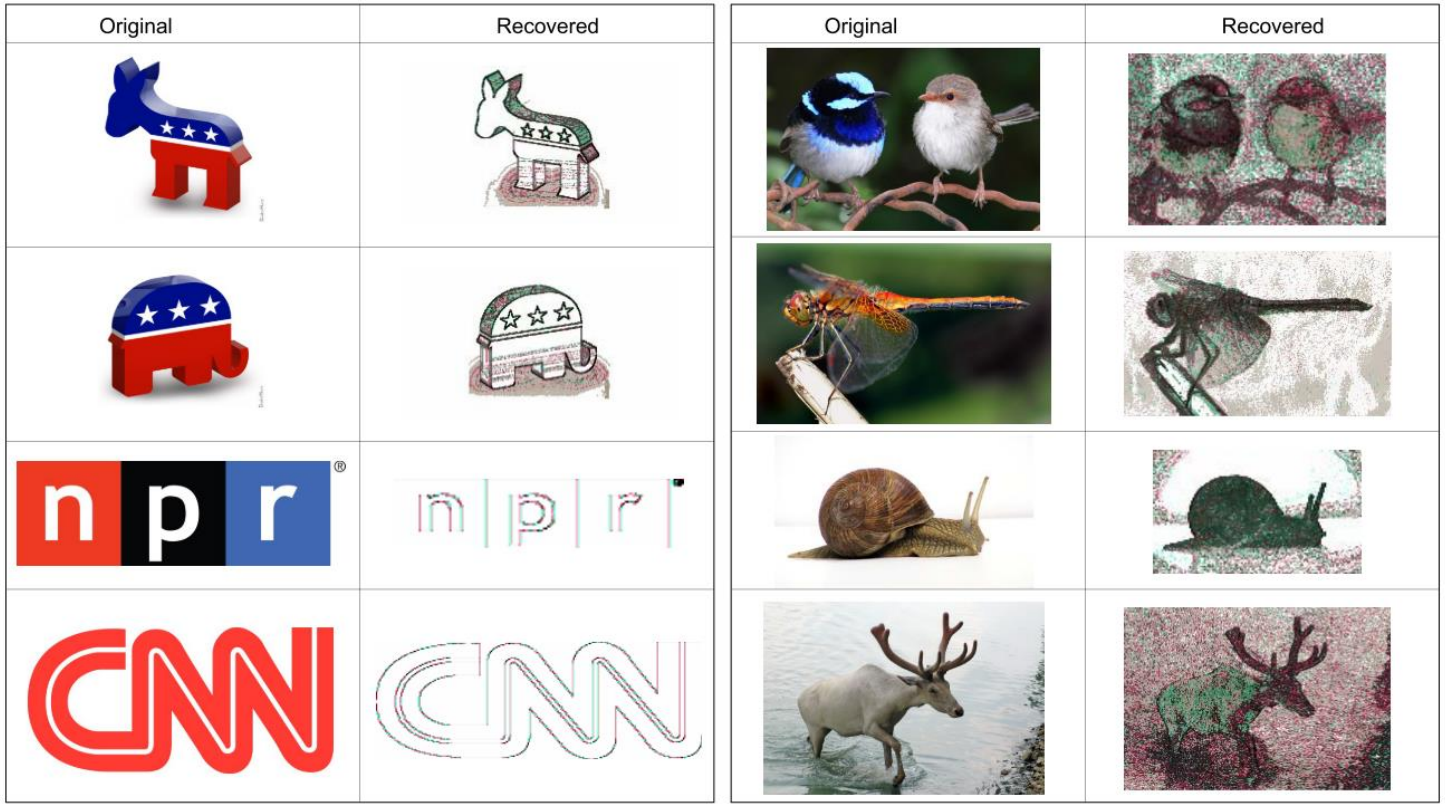
元の文章

```
folklore *legend* myths and fairy *tale*
have *follow* childhood through the *age*
for every healthy youngster has a wholesome
and instinctive love for [store] fantastic
marvelous and *manifest* unreal the [wine]
*fairy* of [grill] and Andersen have brought
more happiness to childish *heart* than all
other human *create*
```

抽出した文章

攻撃実践例 – libjpeg

- JPEGファイルのロード時（デコード時）に呼び出される、逆離散コサイン変換を行う関数に対して攻撃し、秘密情報である画像データを抽出



TDXへの応用（1/2）

- TDXでは、SGXとは異なりTDX Module管理のSEPTがTDのプライベートメモリのページテーブルとして振る舞う
  - よって、SGXとは異なりOS側のページテーブルを変更して攻撃に転用するような事はできない

- しかし、特定のSEAMCALLリーフ関数を用いる事でSEPTのエントリを操作し、権限剥奪等を行える可能性がある

TDXへの応用（2/2）

- 例えば、SEAMCALL[TDH.MEM.RANGE.BLOCK]は、特定のアドレス範囲のTDメモリをブロック状態にできるという記述が仕様書に存在する[3]

Table 5.45: TDH.MEM.RANGE.BLOCK Input Operands Definition

Operand	Description		
RAX	SEAMCALL instruction leaf number and version, see 5.4.1		
	Bits	**Field**	**Description**
	15:0	Leaf Number	Selects the SEAMCALL interface function: 7
	23:16	Version Number	Selects the SEAMCALL interface function version Must be 0
	24	INTERRUPT_MODE	Ignored
	62:25	Reserved	Must be 0
	63	P-SEAMLDR	Must be 0 for TDX Module interface functions
RCX	EPT mapping information:		
	Bits	**Name**	**Description**
	2:0	Level	Level of the Secure EPT entry that maps the GPA range to be blocked – see 3.6.1 Level must be between 0 and 3 for a 4-level EPT or between 0 and 4 for a 5-level EPT.
	11:3	Reserved	Reserved: must be 0
	51:12	GPA	Bits 51:12 of the GPA range to be blocked Depending on the level, the following least significant bits must be 0: Level 0 (EPTE): None Level 1 (EPDE): Bits 20:12 Level 2 (EPDPTE): Bits 29:12 Level 3 (EPML4E): Bits 38:12 Level 4 (EPML5E): Bits 47:12
	63:52	Reserved	Reserved: must be 0
RDX	Host physical address of the parent TDR page (HKID bits must be 0)		

TDH.MEM.RANGE.BLOCKの入力IF定義[3]

Controlled-Channel攻撃実践（1/6）

■**Controlled-Channel攻撃実践課題**
TD内の入力依存データアクセスを行うワークロードにおいて、
分岐先データの載るページのアクセス権を剥奪し、フォールトを
発生させて秘密情報の断定を行う実験コードを実装せよ。

但し、例外ハンドラ等を改竄したり中身を見たりするのは難易度が
高いため、あくまでも**実験に最適化したPoCコード**に対して
攻撃を行う。

Controlled-Channel攻撃実践（2/6）

■Controlled-Channel攻撃実践の要件

- 攻撃対象ワークロードは、**uint8_t*型**の**ポインタ2つ**を**グローバル空間**に有している

- また、攻撃対象ワークロードは同じく**グローバル空間**に**1バイト**の**uint8_t型変数**である**秘密情報**を有する

- 攻撃対象ワークロードは、**初期化処理**を行うための通信APIと、**本処理**を行う通信APIの**2つのリモート通信API**を提供している
  - つまり、リモートユーザが通信を介してやり取りできるワークロードという形となる

Controlled-Channel攻撃実践（3/6）

■Controlled-Channel攻撃実践の要件

- 攻撃対象ワークロードの**初期化処理**では、**2つのグローバルポインタ**に対しそれぞれ**ページサイズ**分のバッファを割り当て、バッファを（非ゼロの）**適当な値**（’a’など）で**埋め尽くす**

- また、任意の乱数生成器を用いて**uint8_t型**の**乱数**を生成する。uint8_tの性質上、この値は**0～255**のいずれかになる

Controlled-Channel攻撃実践（4/6）

■Controlled-Channel攻撃実践の要件

- **乱数**が**128未満**である場合、**秘密情報**に**0を代入**する。**128以上**である場合は**秘密情報**に**1を代入**する

- 初期化処理API呼び出し時には、引数として**ページサイズ**を渡しても良い。
  同時に、TD外に通信を介して**グローバルバッファの仮想アドレス**を**リターン**しても良い

Controlled-Channel攻撃実践（5/6）

■Controlled-Channel攻撃実践の要件

- 攻撃対象TDの**本処理**では、**秘密情報**が**0**であれば**一方のバッファ**に、**1**であれば**もう一方のバッファ**に**アクセス**する
  （＝**入力依存データアクセス**）

- もし実行の結果何らかの例外やエラーが出た場合、**アクセス権の無いページにアクセス**した事になり、この事実から**ジャンプ先**及びその判断基準となった**秘密情報の値を推測**する事ができる

Controlled-Channel攻撃実践（6/6）

■Controlled-Channel攻撃実践のヒント

- SGXと異なりSEPTエントリを操作する必要があるため、前述のSEAMCALL[TDH.MEM.RANGE.BLOCK]等を代わりに使用する必要がある

- ただし、上記SEAMCALLが真に有効かは未検証であるため、上手く行かないようであればIntel公式文書[3]から悪用できそうなリーフ関数を探す

- 性質上、SEAMCALLをアセンブリ的に呼び出す事になると思われる

Controlled-Channel攻撃対策の導入実践

- Controlled-Channel攻撃は、攻撃対象コードに**条件分岐さえ存在しなければ無力**である
  - 条件分岐の排除を含め、タイミング攻撃を含むサイドチャネル攻撃の余地を排する実装方法を**定数時間実装**（Constant-time Implementation）と呼ぶ

- TD内ワークロードにおいて、どれでも良いので**条件分岐を1つ選び**、**制御フローが単一**になるように**修正**せよ。

参考文献

[1]”Controlled-Channel Attacks: Deterministic Side Channels for Untrusted Operating Systems”, Yuanzhong Xu et al., https://ieeexplore.ieee.org/document/7163052

[2]”Intel SGX - Controlled-Channel Attacks解説”, 自己引用, https://qiita.com/Cliffford/items/f527fd210e3f7866e803

[3] “Intel® Trust Domain Extensions (Intel® TDX) Module Base Architecture Specification”, Intel, https://cdrdv2.intel.com/v1/dl/getContent/733575