猫でもできる！
ふつうの脆弱性ハンティング

丹田 賢

バリバリの

オフェンス側

の視点です^^

要旨

- ゴールは皆さんが脆弱性を発見できるようになること
- 「脆弱性ハンティング＝難しい」は誤り
- 脆弱性を探すには、大きな権限での入力値の処理部分を狙う
- IOCTLはその代表、2/6は脆弱性あり！

自己紹介

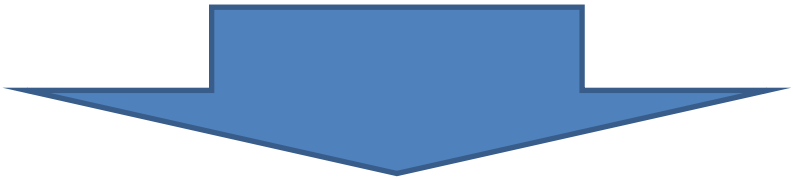
- 丹田賢（tanda.sat@gmail.com）
- 5年くらいWindowsのカーネルで遊んでます
- 3年ぐらいセキュリティっぽい研究開発のお仕事をしています
- お仕事で技術者向けにセキュリティ教育をすることもあります
  - 教材作ったり、教えたり

今回扱う事例

どちらもデバイスドライバーに脆弱性があり、
本来は起こしえないSTOPエラー（ブルースクリーン）が起こせる

このセッションの目的

- 脆弱性ハンティングの戦略・プロセスを紹介



- 同種の脆弱性を発見できるようになる！
- どのようにして脆弱性を探すのかに着目

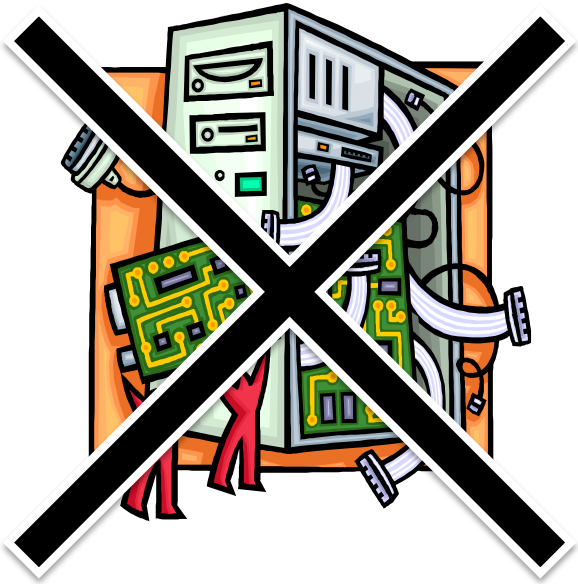
脆弱性を探すメリット

- カッコイイ！！
- 名前が載る！
- 技術力の証明になることがある
- 脆弱性を作り込みにくくなる
- 焼肉に変える

この脆弱性情報は、情報セキュリティ早期警戒パートナーシップに基づき下記の方が IPA に報告し、JPCERT/CC が開発者との調整を行いました。
報告者: 株式会社フォティーンフォティ技術研究所 丹田 賢 氏

「ふつうの脆弱性ハント」の特徴

- アセンブラ読みません
- デバッガ使いません
- 特定のフォーマットや技術仕様に関する知識を要求しません



目次

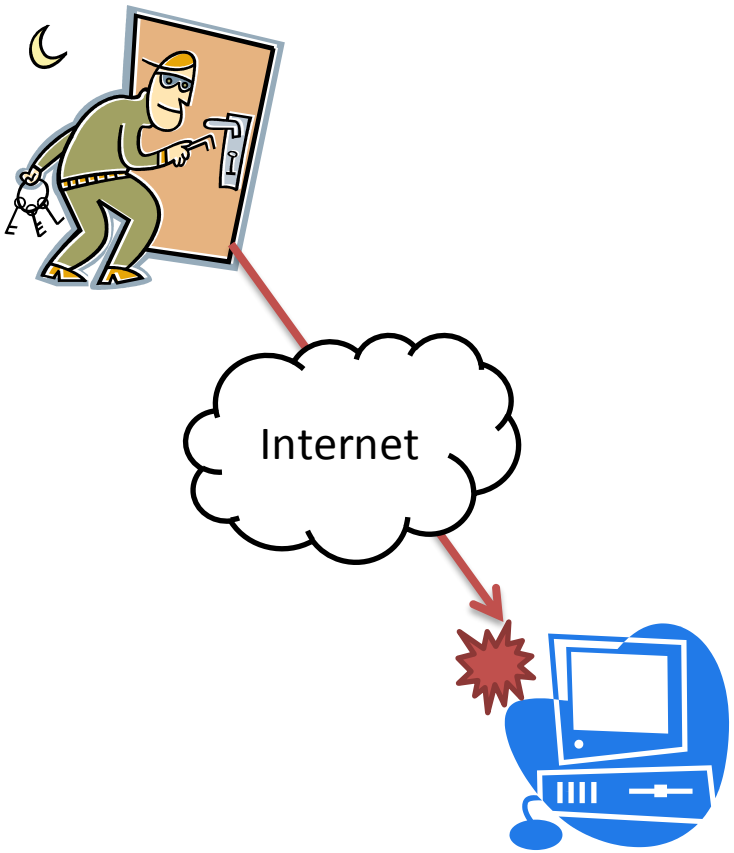
- どこを狙うのか
  - なにが脆弱性になのか
  - どこに脆弱性が作り込まれやすいのか
- 事例分析
  - IOCTLとは
  - どのようにして脆弱性を発見したか
  - 届出から修正完了報告までの流れ
  - 脆弱性の予防策（開発者向け）
- まとめ

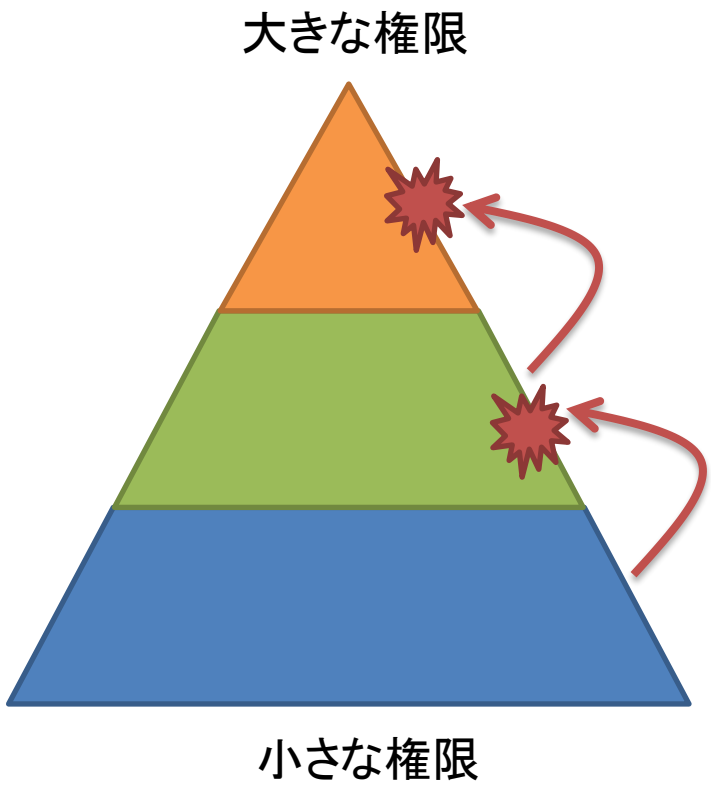
目次

- どこを狙うのか
  - なにが脆弱性になのか
  - どこに脆弱性が作り込まれやすいのか
- 事例分析
  - IOCTLとは
  - どのようにして脆弱性を発見したか
  - 届出から修正完了報告までの流れ
  - 脆弱性の予防策（開発者向け）
- まとめ

なにが脆弱性なのか

- 本来出来ないはずのことができる



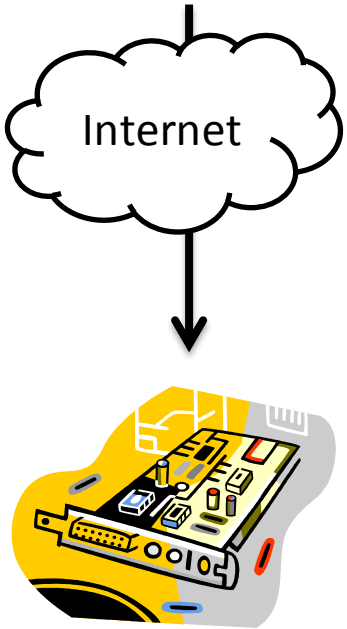


どこに脆弱性が作り込まれやすいのか

- 入力値の処理

ファイルパース

NWからの入力



引数の検証

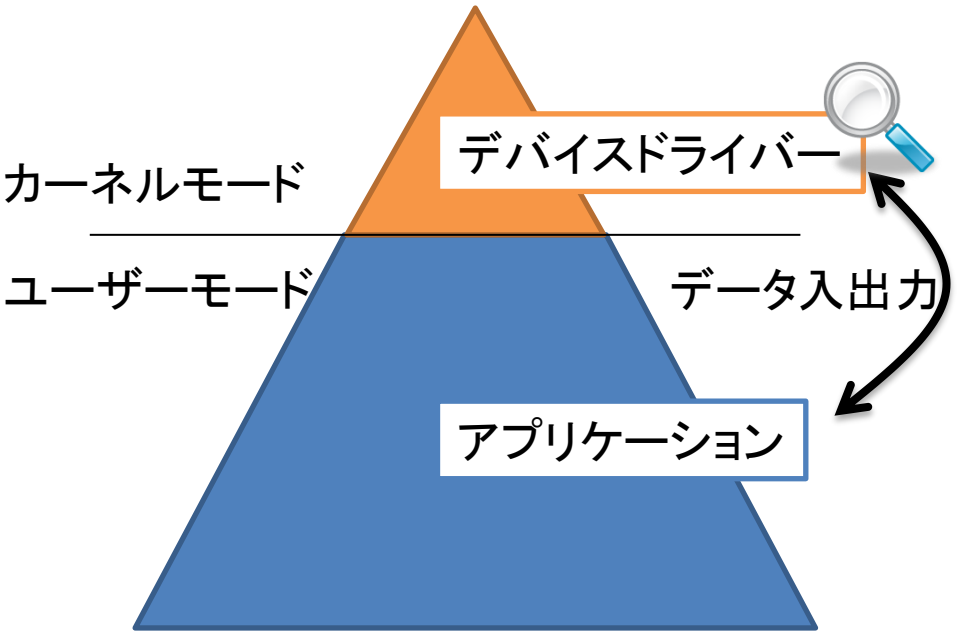
int sum(xxx, …);

目次

- どこを狙うのか
  - なにが脆弱性になのか
  - どこに脆弱性が作り込まれやすいのか
- 事例分析
  - IOCTLとは
  - どのようにして脆弱性を発見したか
  - 届出から修正完了報告までの流れ
  - 脆弱性の予防策（開発者向け）
- まとめ

IOCTLとは

- 脆弱性の宝庫です(*´д`*)



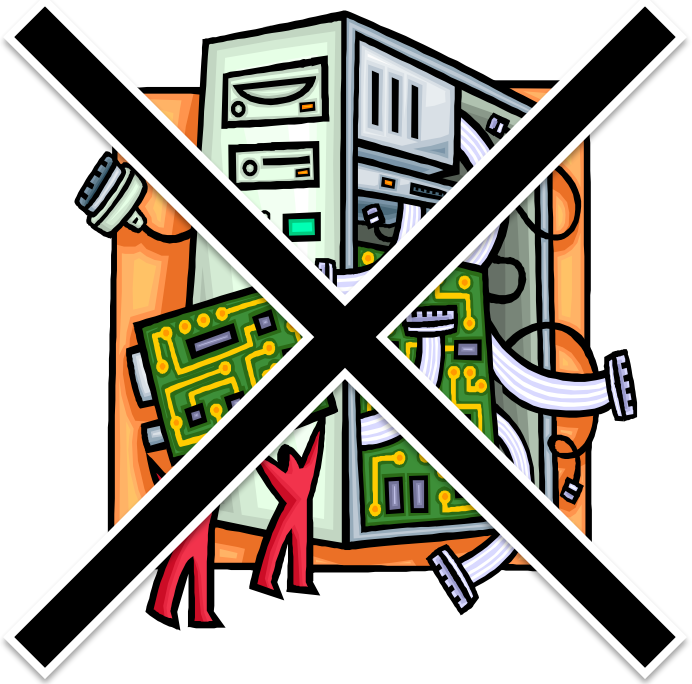
- ✓ 入力値の検証がより大きな権限(カーネル)で行われる
- ✓ 引数検証の実装が難しい(複雑化)

どのようにして脆弱性を発見したか

- VM にドライバーを使っていそうな製品をインストール
- Process Exprolorerでインストールされたドライバーを確認
- WinObjでデバイス名とアクセス権を確認
  - 小さい権限（RISTRECTED=Guest)でも操作できるか
- 拙作IoCtlMonitorを使って使用されているIOCTLを確認
  - http://p.tl/RHm9
- 適当にファザーを作ってファジング
- BSODしたらBSODさせたデータを調べてExploitを作成

ふつうの脆弱性ハントのプロセス（再）

- アセンブラ読みません
- デバッガ使いません



ね、簡単でしょ？

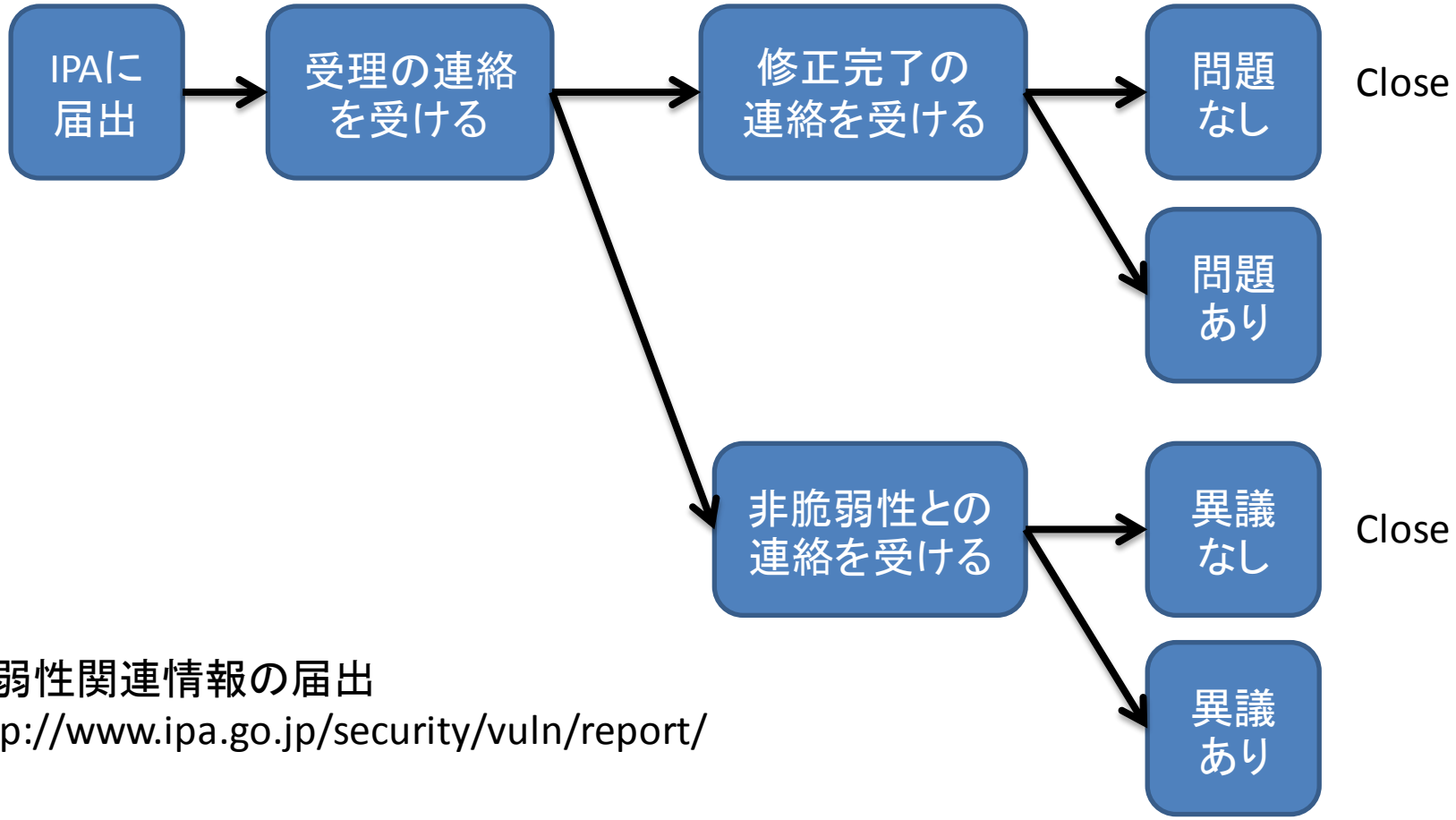
脆弱性の発見は難しい
とは限らない！

有効性は高い

- 前述のプロセスで以下の製品をファジング
  成功率は2/6

製品名	結果
AVG Anti-Virus	発見できず
Avast!	発見できず
KINGSOFT Internet Security	BSOD!!
FFR yarai	発見できず
MagicDisc	発見できず
Daemon Tools シリーズ	BSOD!!

届出から修正完了報告までの流れ



脆弱性関連情報の届出
http://www.ipa.go.jp/security/vuln/report/

脆弱性の予防策（開発者向け）

- 原因
  - 仮想ドライブソフト（Daemon Tools）
    - IOCTL実行にアクセス制御がなく、IOCTLで渡されたデータの妥当性を検査していない
  - セキュリティソフト（Kingsoft）
    - IOCTL実行にアクセス制御が甘く、IOCTLで渡されたデータの妥当性の検証処理そのものが安全でない

脆弱性の予防策（開発者向け）

- 具体的な問題点
  - ドライバがIoCreateDeviceを使ってデバイスを作成していたため、Guest権限でもReadOnly（GENERIC_READ）でデバイスハンドルを取得することができた
  - すべてのIOCTLのAccessにFILE_ANY_ACCESSまたはFILE_READ_ACCESSが指定されていたため、ReadOnly権限のデバイスハンドルでIOCTLが実行できた
  - 「正しいIOCTL_CODEに正しくないデータ」が渡されることを想定していない、または検査が安全でないために、IOCTLのデータがカーネル空間で問題を起こした

脆弱性の予防策(開発者向け)

- ベストプラクティス
  - 権限の低いユーザーからのデバイス操作が不要なのであればIoCreateDeviceではなくIoCreateDeviceSecureを使用する
    - たとえば、SDDL_DEVOBJ_SYS_ALL_ADM_ALLを指定すると、Admin権限以外からのデバイスハンドルの取得を完全に制限できる
  - IOCTLにアクセス制御を適用するためにCTL_CODEのAccessにFILE_ANY_ACCESSを指定しない
    - IoCreateDeviceを使った場合、EveryoneにはReadWriteが与えられるのでこれはIoCreateDeviceSecureほどには効果的ではない。
  - IOCTLに渡されたデータを信頼しない(検証する)

脆弱性の予防策（開発者向け）

- リソース
  - FILE_ANY_ACCESSの問題
    - en:http://msdn.microsoft.com/en-us/windows/hardware/gg463225
  - 信頼性、セキュリティ、およびメンテナンス
    - en:http://msdn.microsoft.com/en-us/windows/hardware/gg487478
    - ja:http://msdn.microsoft.com/ja-jp/windows/hardware/gg487478
  - WinObj（デバイスにかけられたアクセス制御を確認する）
    - http://technet.microsoft.com/en-us/sysinternals/bb896657
  - ドライバーの信頼性に関する一般的な問題
    - en:http://msdn.microsoft.com/en-us/windows/hardware/gg487311

まとめ

- 脆弱性見つけるとカッコイイ！名前も載る！
- 脆弱性ハンティングには難しくないものもある
- 脆弱性を探すには、大きな権限での入力値の処理部分を狙う
- IOCTLはその代表、2/6は脆弱性あり！

ありがとうございました！

Q&A

- Q:ツール公開してないんですか？
- A:モニターは公開済みです。ファザーは業務コードが混じってたので公開できません☹

Q&A

- Q:ユーザーモードからカーネルのファジングって大変じゃない？
- A:カーネルが口あけてるところにポンポン値投げ込むだけなんで余裕。

memo

- ファジングはかならずGuestで行ってください
- “一般ユーザー”は“本来BSODが起こせます”
  - NtRaiseHardErrorはそういういう“機能”があります
  - 正確にはSeShutdownPrivilegeを持つ場合可能
- そのため“本来BSODが起こせない”Guestでファジングしなければなりません
- Good luck!!