軽量化を図ったランサムウェア
特化型ディフェンダー:

ランサムウェアワクチン

研究駆動コース 38R 守田向輝

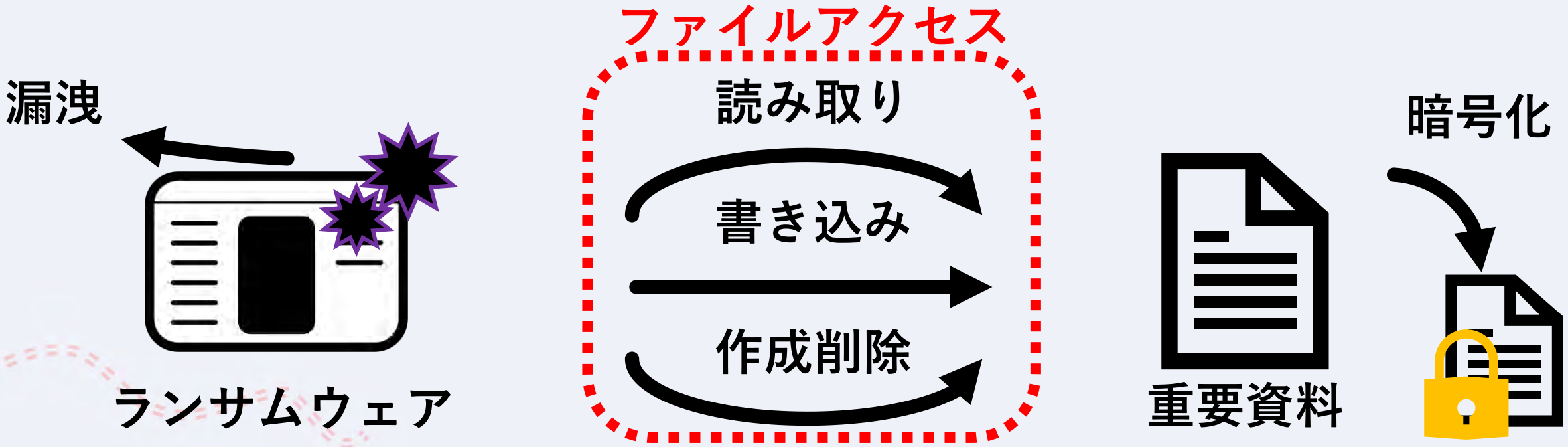
背景

- 信頼のできないドメインのwebやメールからダウンロードした実行ファイルが、近年脅威(アサヒグループやアスクル等の事件)を増すランサムウェアの可能性がある

- 従来のアンチウイルスソフト製品では多くのファイルをスキャン、または動的プログラムの解析などをするため、リソースを多く消費する
  - 以前クラスメイトの学校用PCにランサムウェアが感染したことがあった
  - 学校用PCはスペックが低いうえに生徒の操作権限が少ないためWinDefなどの設定の際に躓いた
    →**管理者権限のいらない軽量なディフェンダーが作れないか**

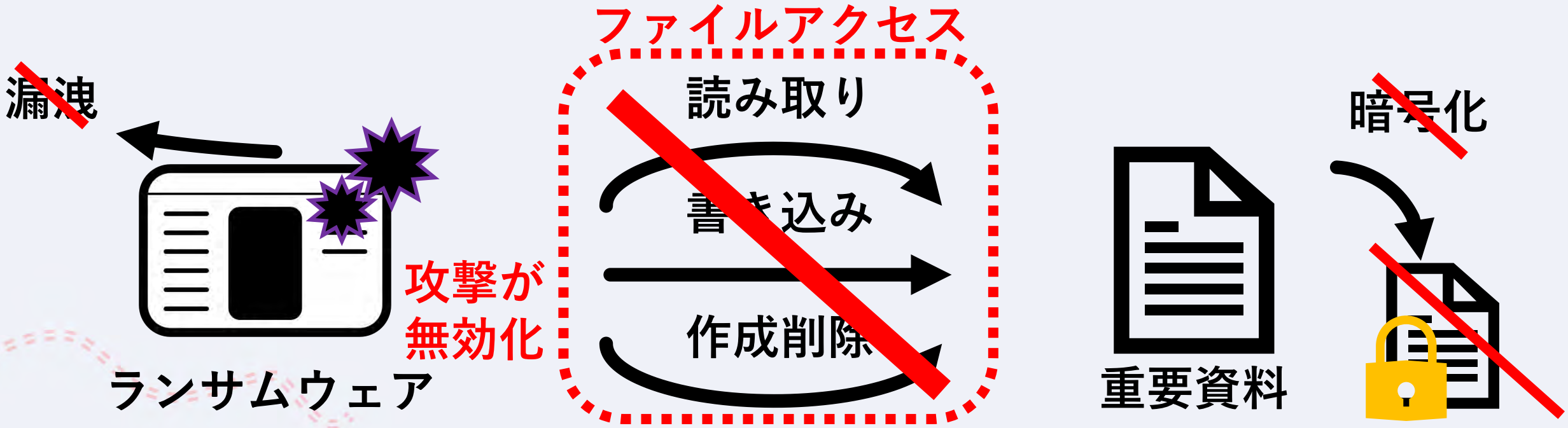
防御対象とするランサムウェアの特徴

- ランサムウェアはユーザーの多くのファイルを暗号化、窃取するマルウェア
- ランサムウェアが暗号化や窃取を行う際、多くのランサムウェアは
  ユーザーのファイルへ**ファイルアクセス**をほぼ必ず行う



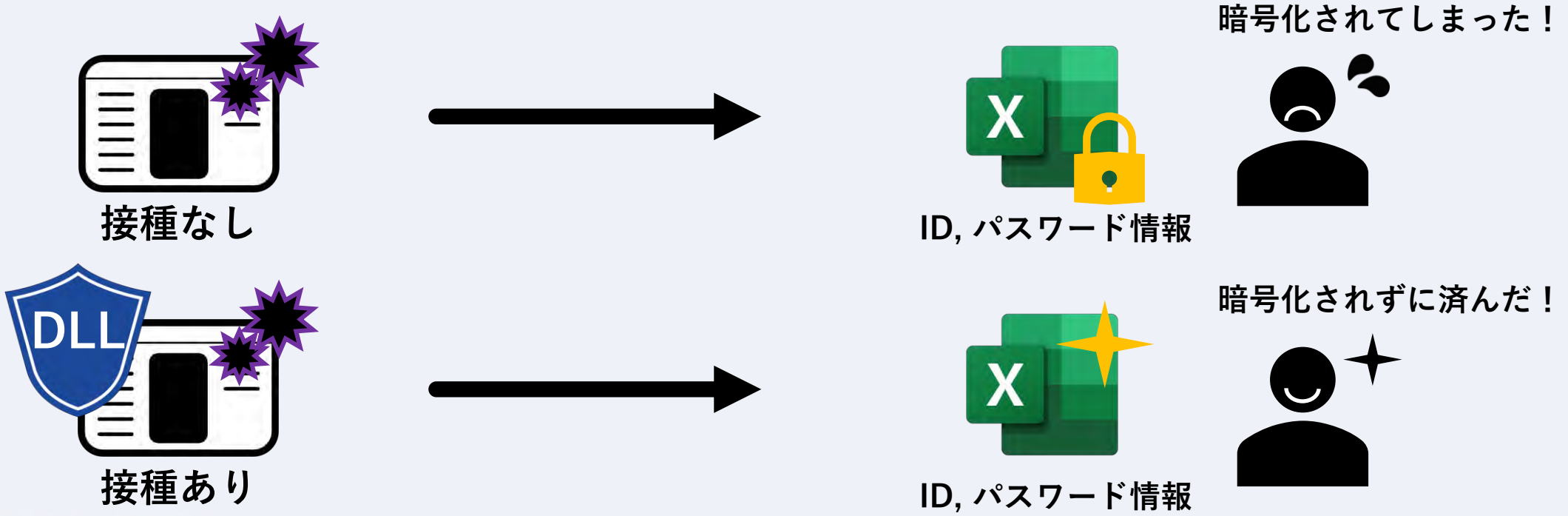
提案手法:ランサムウェアワクチン

- 防御手法:特定の実行ファイルが行う**ファイルアクセス**を自作ディフェンダーで無条件なプログラムで**ブロック**する
- 省リソース化:複雑な処理を行わずに、攻撃を軽量に、かつ簡易的にブロックする
- 利用する技術:
  ファイルアクセスをブロックする手法として**IAT Hook**を活用
  IAT Hookを行う関数を読み込ませる手法として、**DLL Injection**を活用

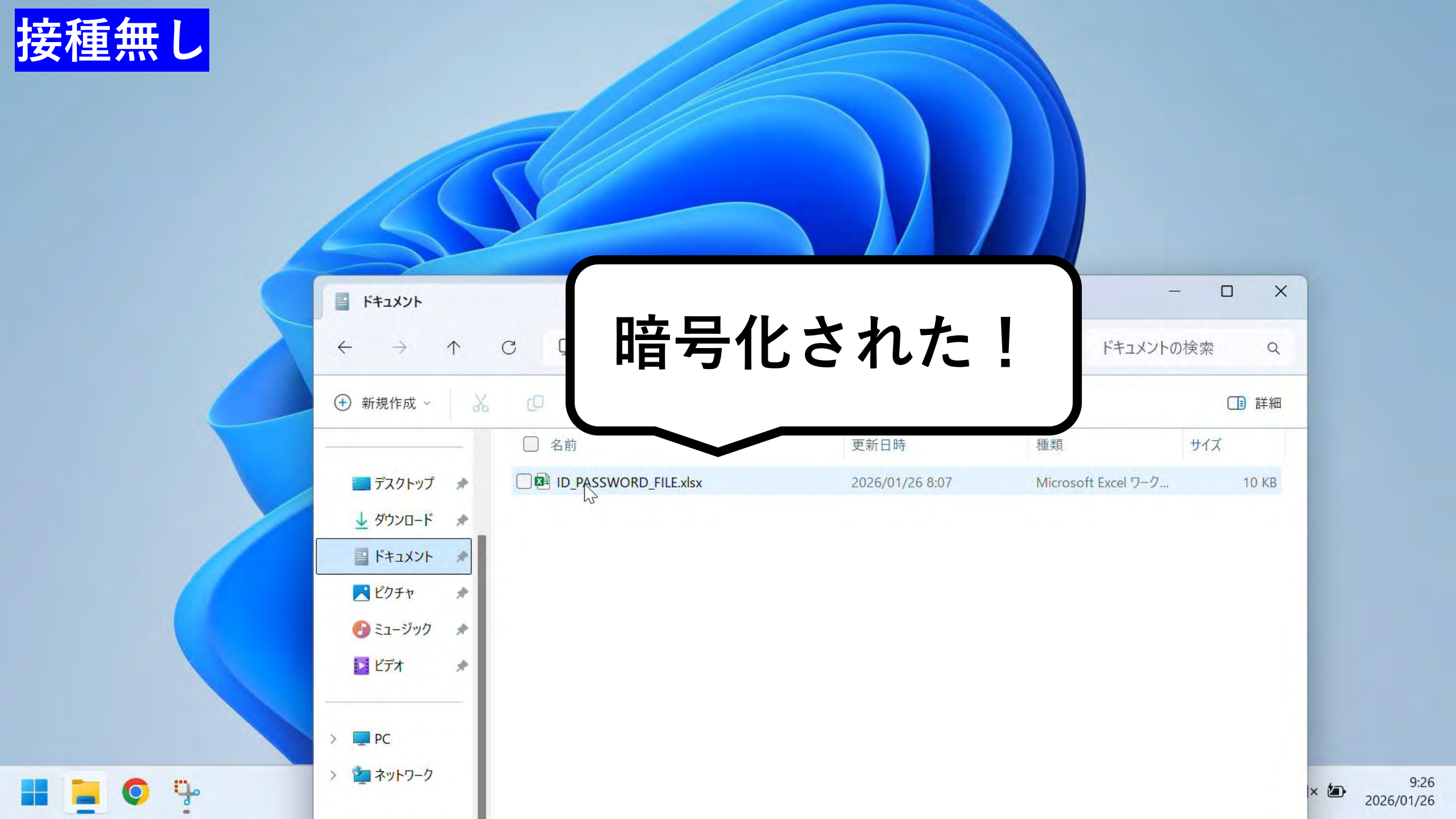


デモ

- ワクチン接種無とワクチン接種有のサンプルファイルの被害状況を比較
  - ここでのサンプルファイルはExcelで管理しているID, パスワード情報とする
  - Excelファイルはユーザーフォルダ(Documents)に配置



接種無し


ドキュメント
ドキュメントの検索
新規作成
詳細
暗号化された！
名前
更新日時
種類
サイズ
ID_PASSWORD_FILE.xlsx
2026/01/26 8:07
Microsoft Excel ワーク...
10 KB
デスクトップ
ダウンロード
ドキュメント
ピクチャ
ミュージック
ビデオ
PC
ネットワーク
9:26
2026/01/26

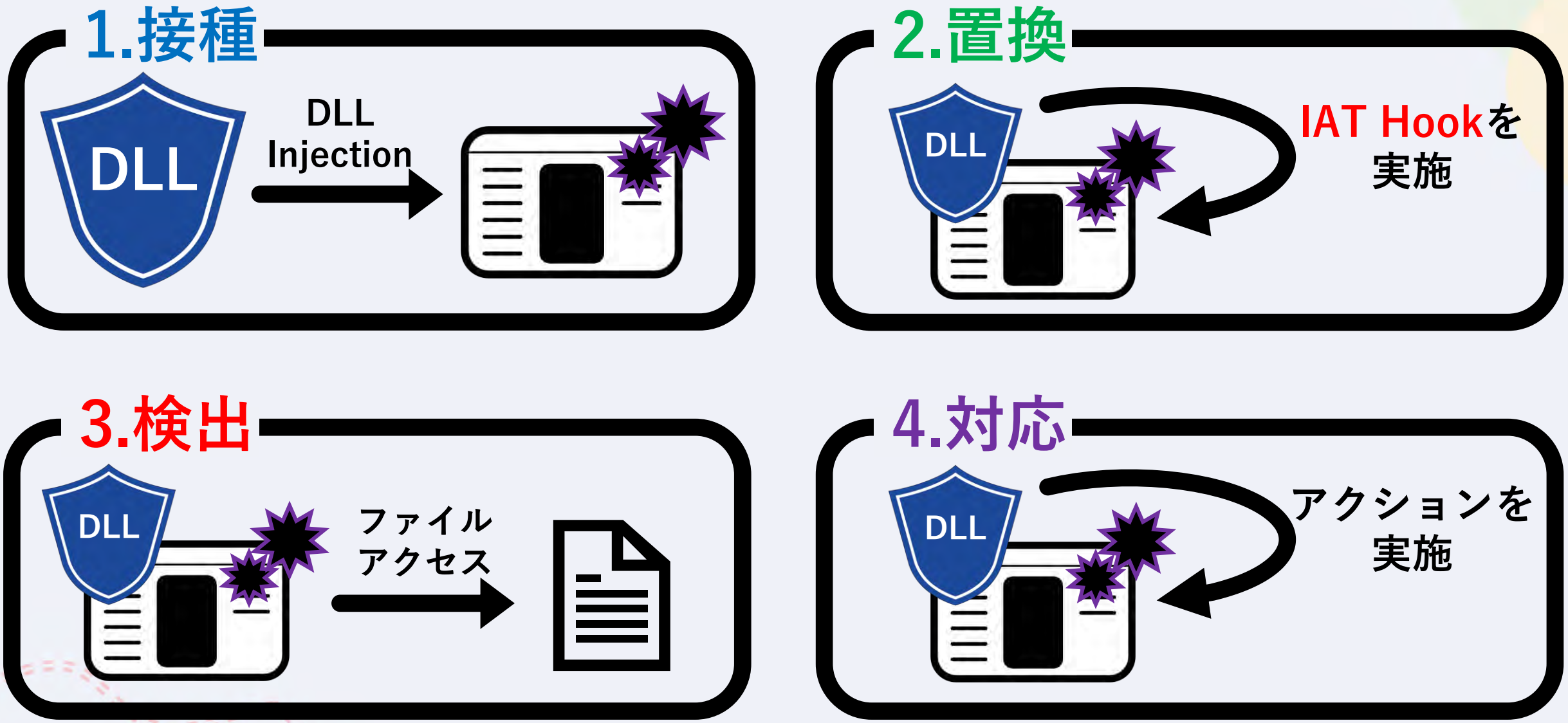
接種有り

ws PowerShell

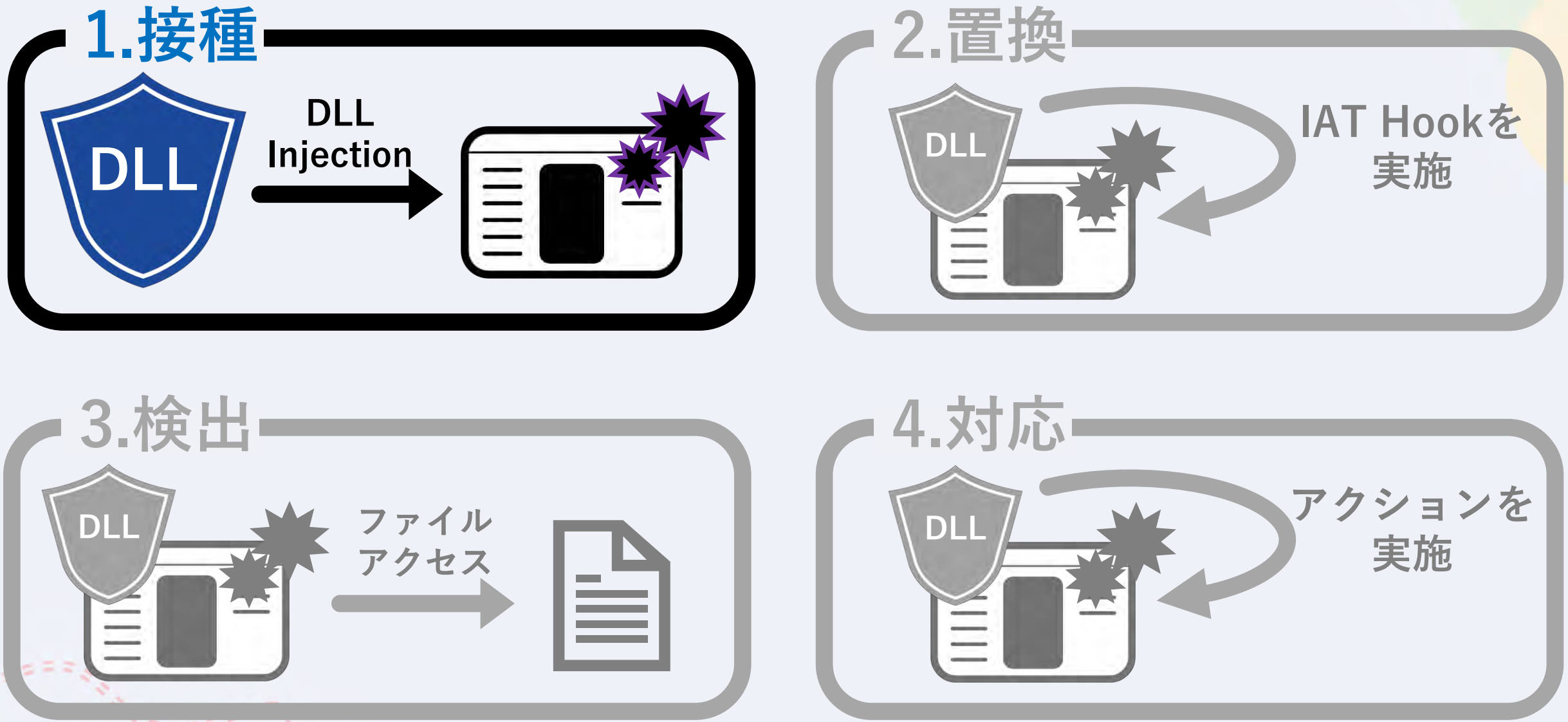
```
ers\C22155\Downloads> ./ransomwarevaccine_injection.exe RANSOMWARE.exe -WDB -TC -F C:/Users/C22155/Downloads
```

暗号化されてない！

開発したランサムウェアワクチンの流れ

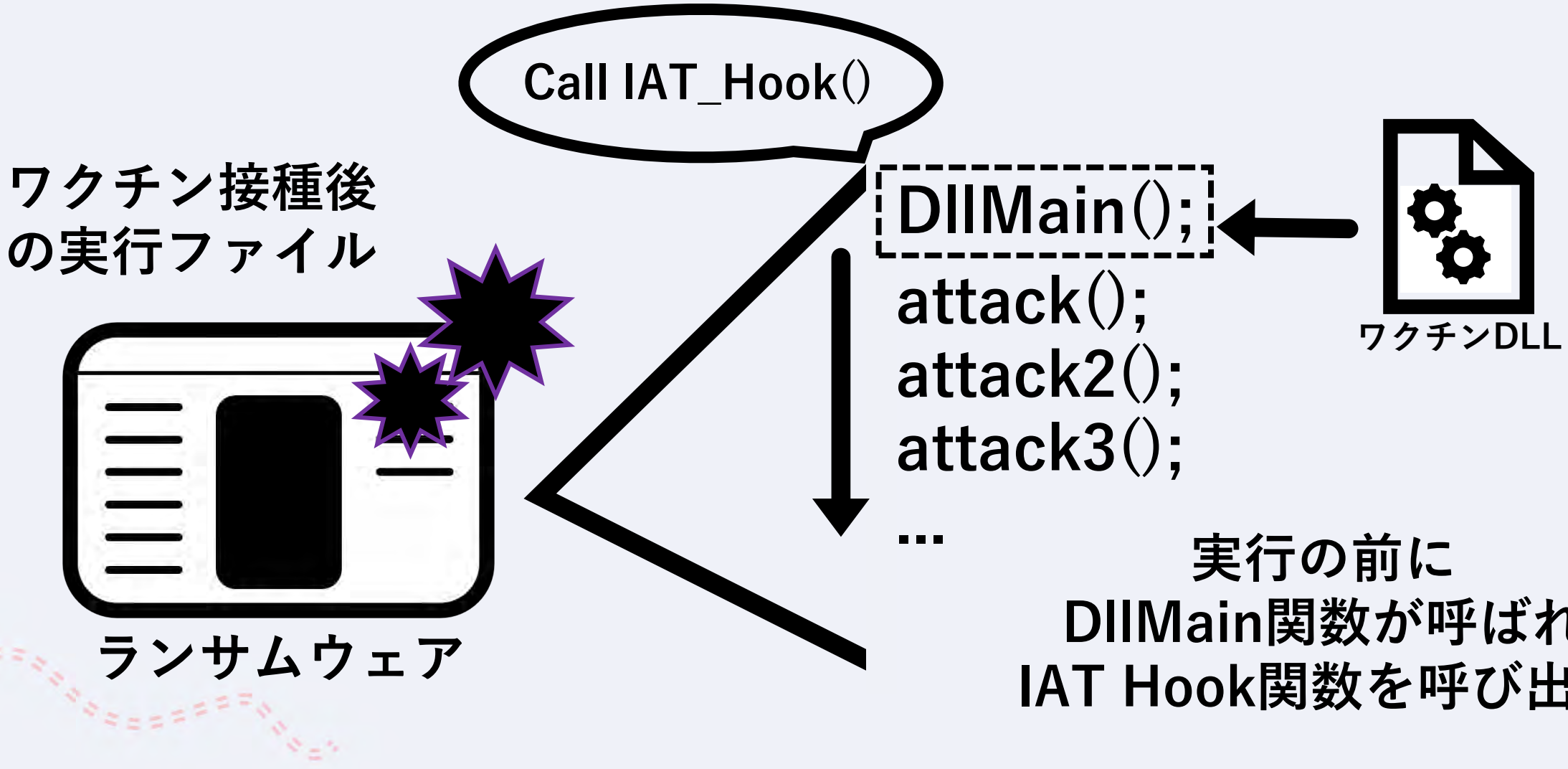


開発したランサムウェアワクチンの流れ



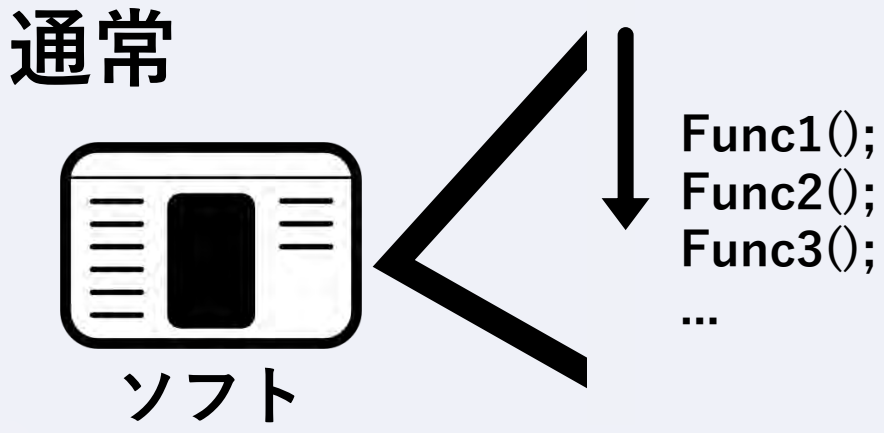
1.接種

- ランサムウェアが悪意のあるプログラムを実行する前に**DLL Injectionを実施**
  - ランサムウェアのプログラムの前にDllMain関数内がIAT Hook関数を呼び出し

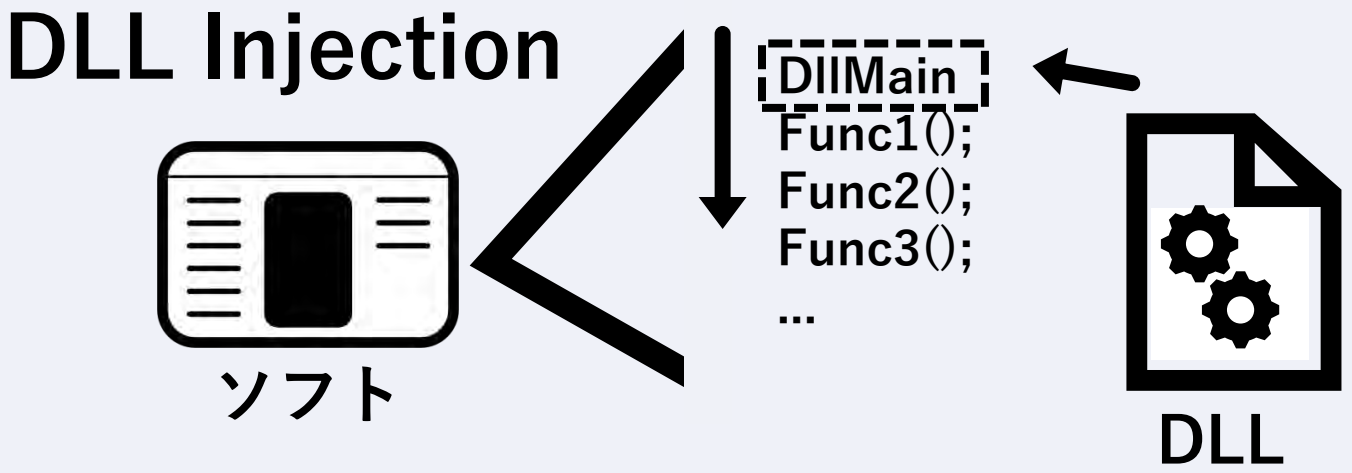


1.接種:DLL Injectionとは

- DLLが読み込まれた時に、自動で呼ばれるDllMain関数という関数をDLLに組み込む
- そのDllMain関数に任意のコードを書いたDLLを、強制的にソフトに読み込ませる
- ソフトに読み込まれるとDllMain関数が起動、**ソフト内で任意のコードが実行**される

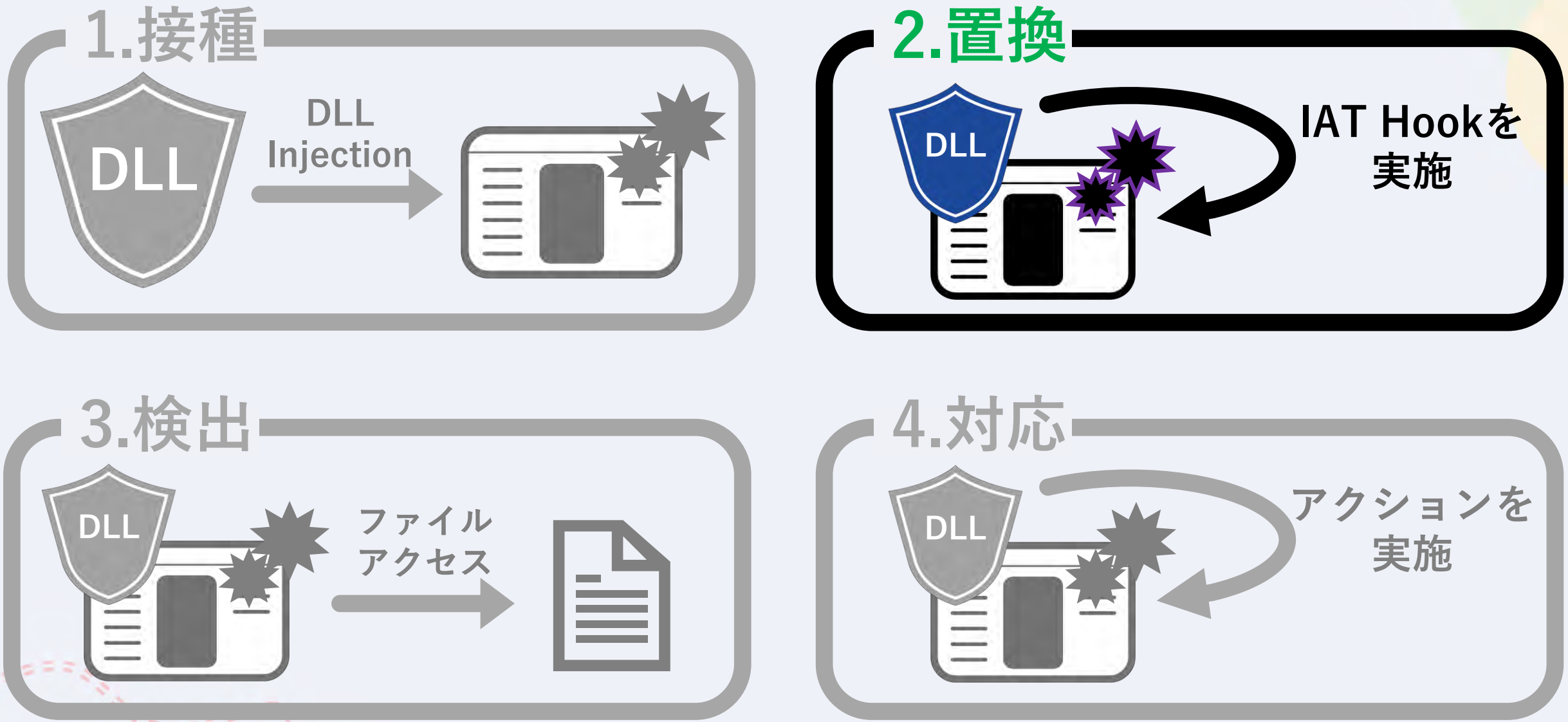


ソフトは事前に組み込まれた
プログラム通りに動く



ソフトが動く前に、ソフト内で
DllMain(任意のコード)を実行できる

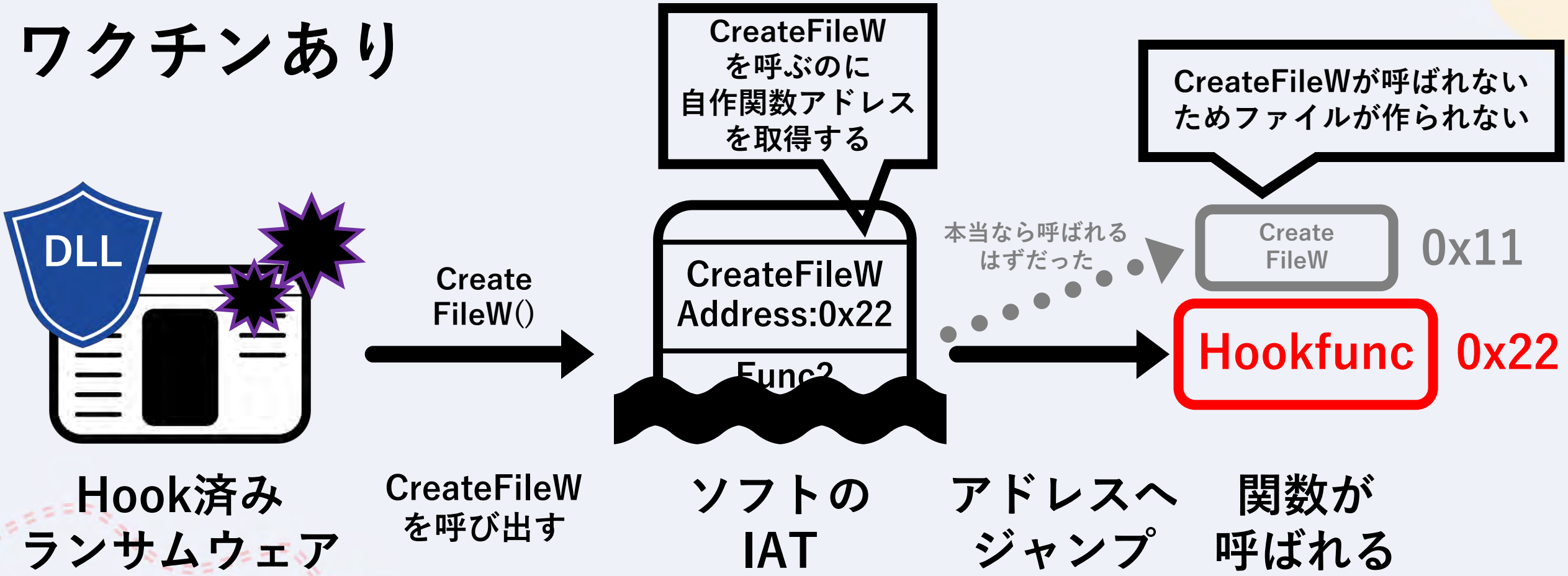
開発したランサムウェアワクチンの流れ



2.置換

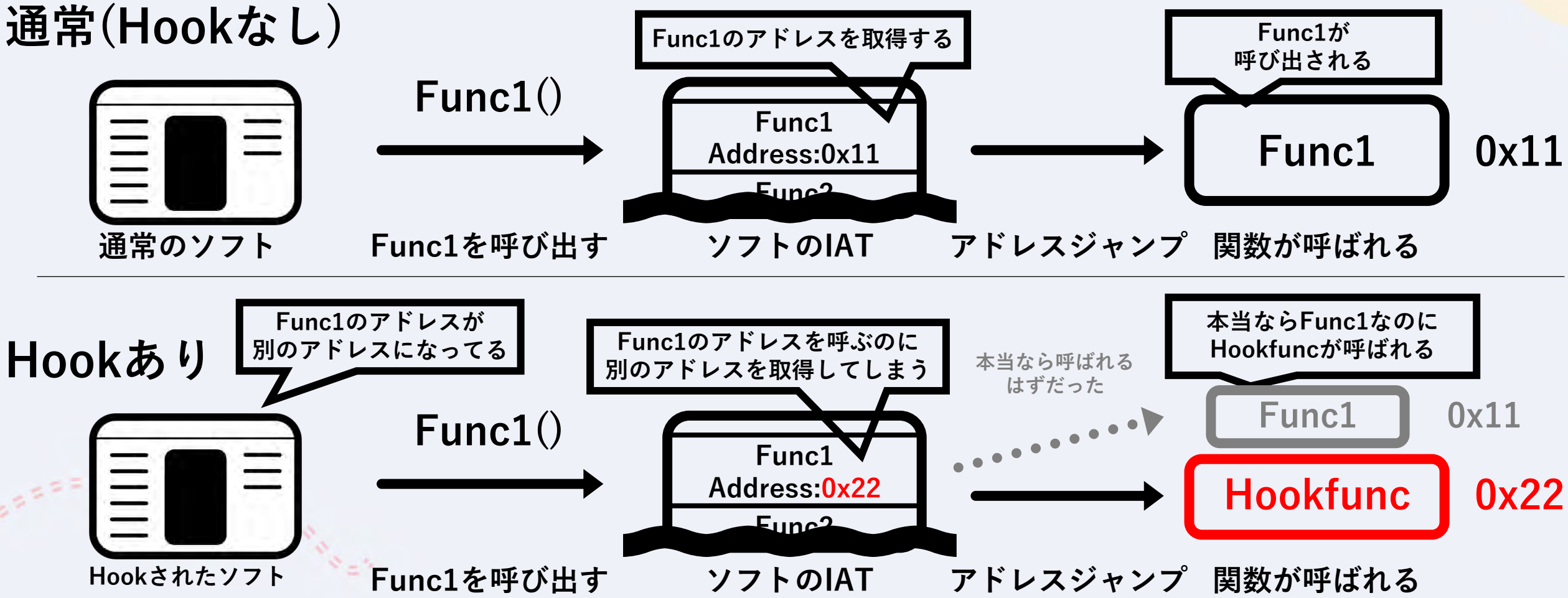
- ファイルアクセス関数(例:CreateFileW)をIAT Hookする
  - ランサムウェアがファイルアクセスを行っても**自作関数(Hookfunc)**が呼ばれる
- ファイルアクセス関数(例:CreateFileW)が呼ばれないためファイルが作られない

ワクチンあり

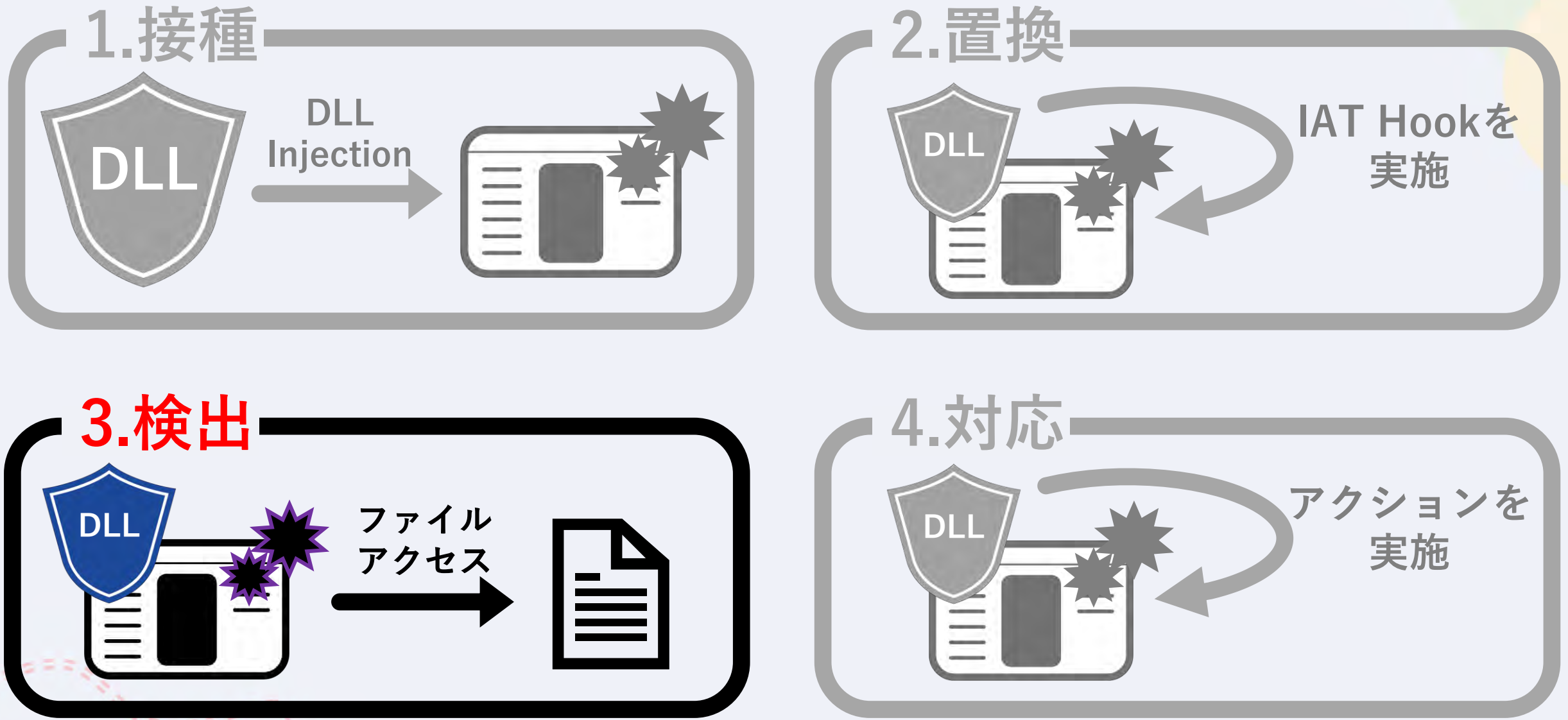


2.置換:IAT Hookとは

- 実行ファイルにはIATという様々な関数のアドレスが保存されている空間がある
- その中の特定の関数のアドレスを別の関数のアドレスに編集することにより
  特定の関数が呼ばれると別の関数を呼ぶことが出来る

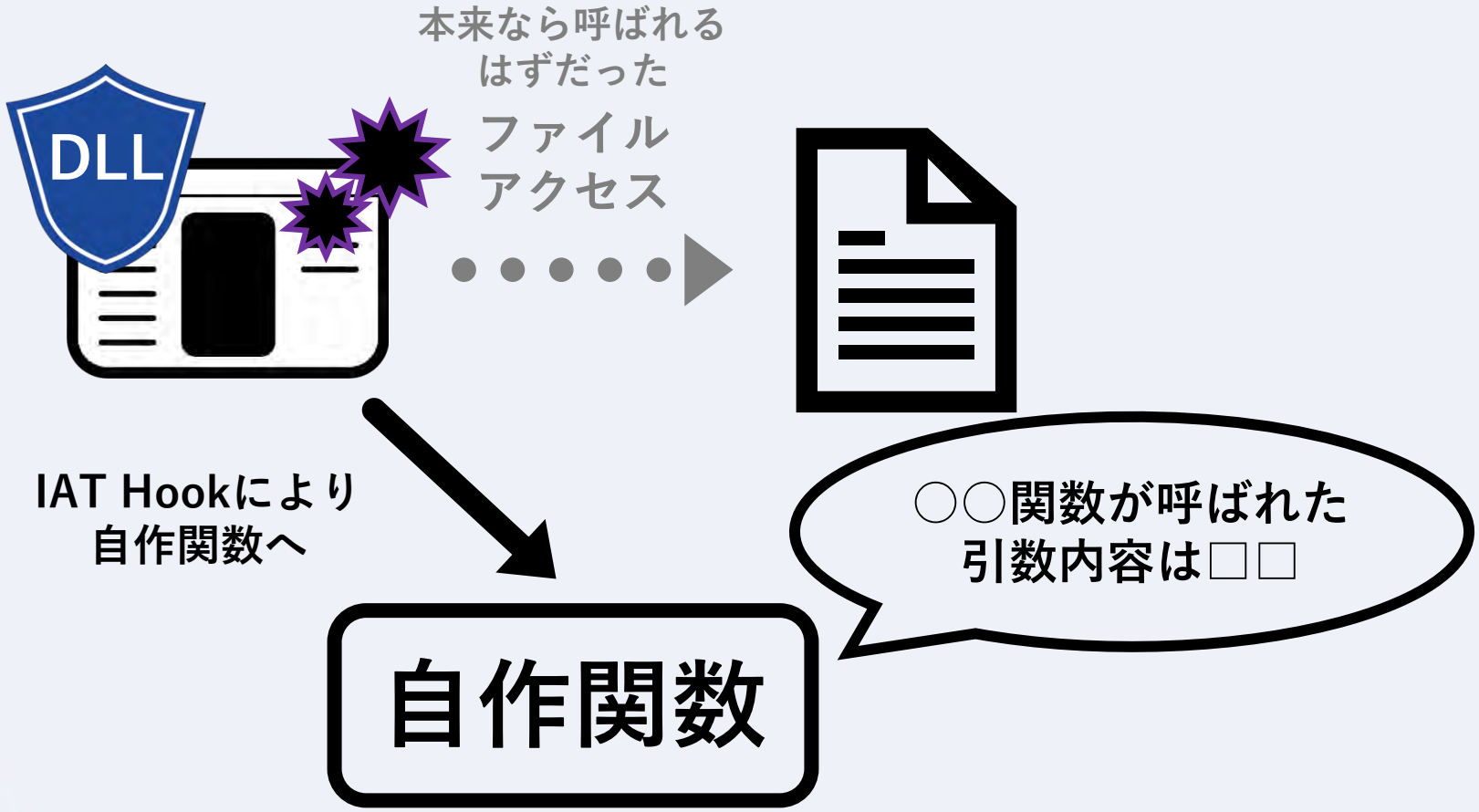


開発したランサムウェアワクチンの流れ

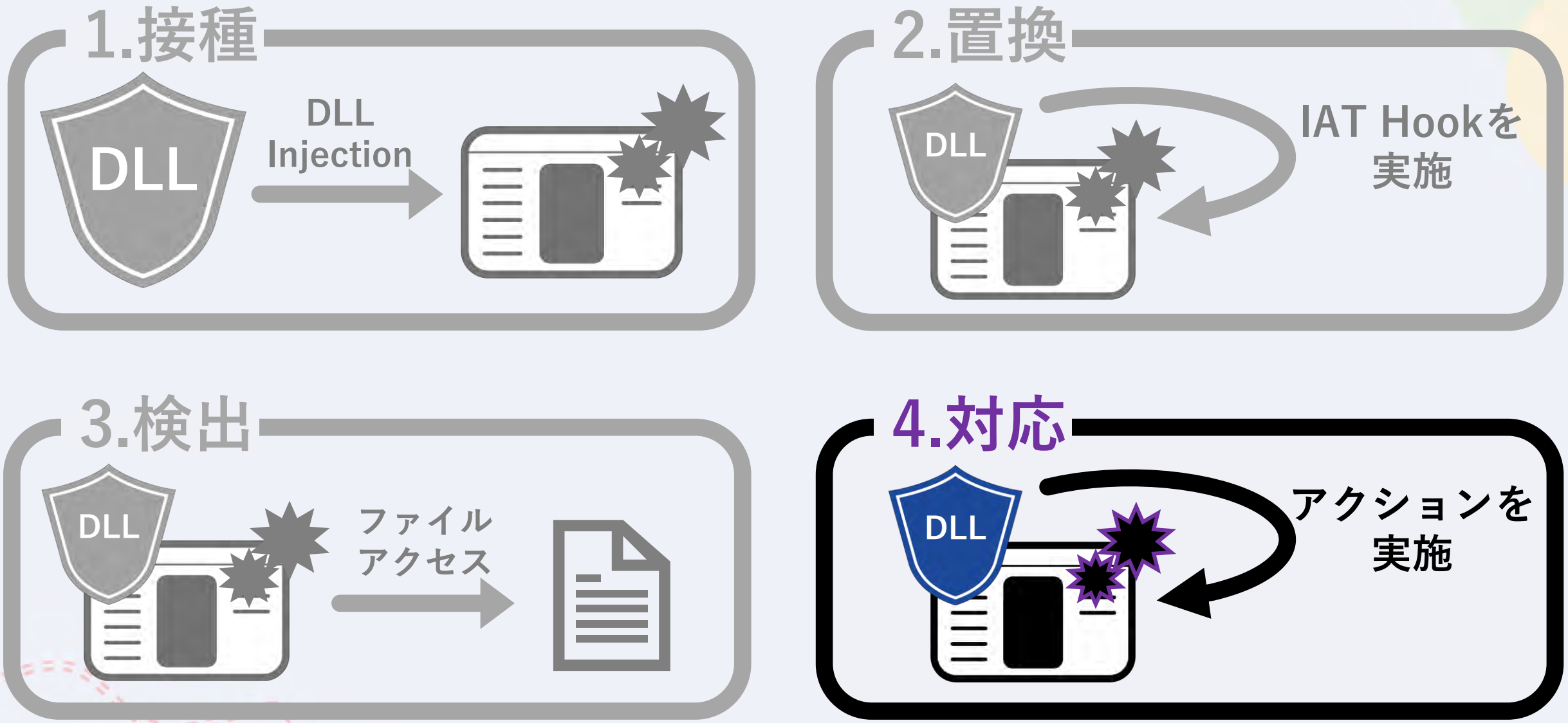


3.検出

- IAT Hookにより各ファイルアクセス関数に対応された自作関数が呼び出し
- 本来呼ばれるはずだった関数への引数内容も自作関数が取得
  - それらの情報を元に自作関数が対象ファイルと操作内容を検出

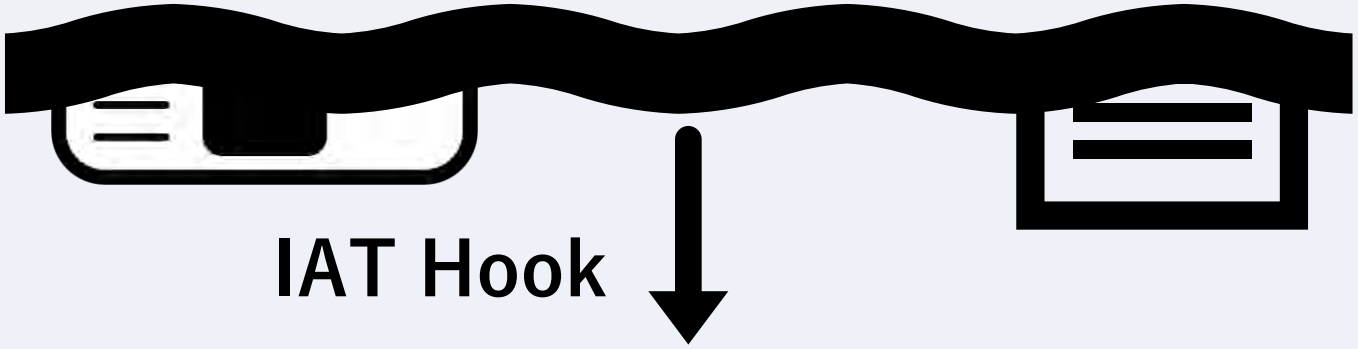


開発したランサムウェアワクチンの流れ



4.対応

- ユーザーの設定(オプション)と3.検出で得た情報を元に**自作関数がアクションを実行**
  - 元のファイルアクセス関数を呼び出すor呼び出さないなどの実行の制御
  - 対象ファイルや操作内容の通知



IAT Hook

自作関数

判定のオプションが AB(全てブロック) → **関数をブロック**

通知のオプションが TC(CUIで通知) → **関数名と引数 内容を通知**

etc...

ブロック例(自作関数の流れ)

元の関数, 引数を保存

(AN:全てのアクセスを許可)

オプションがABだった → **元の関数を呼ばない！**

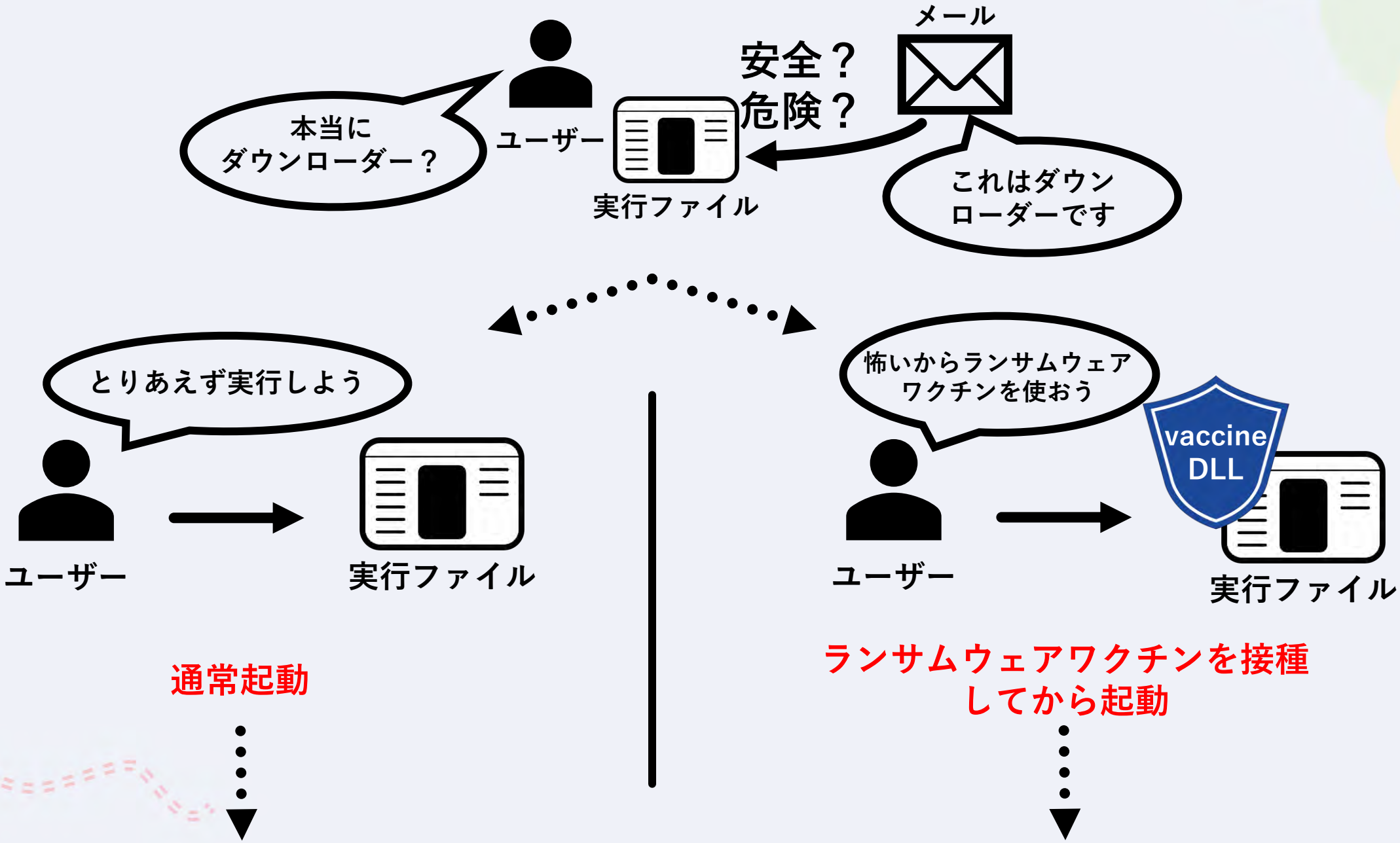
オプションがANだった → **元の関数を呼ぶ！**

etc...

通知例

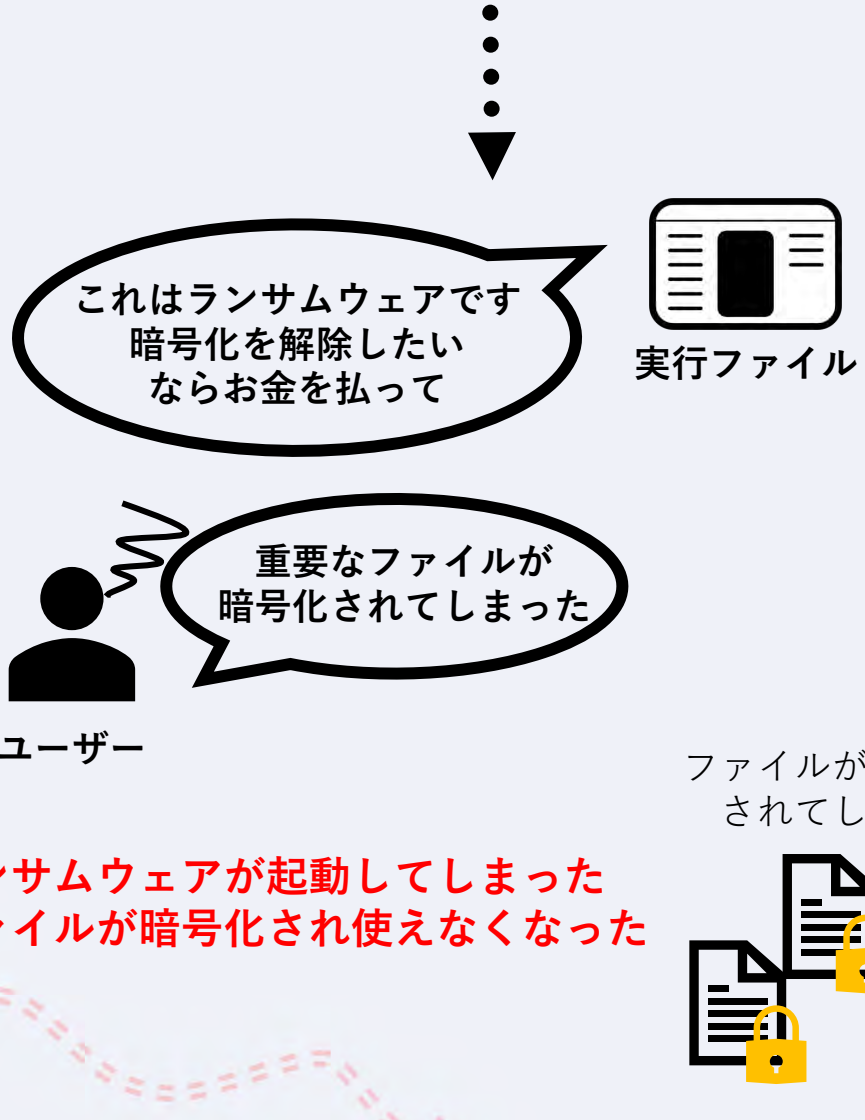
```
WriteFile for \\?\C:\Users\C22155\Documents\ID_PASSWORD_FILE.xlsx.Locked
ReadFile for \\?\C:\Users\C22155\Documents\ID_PASSWORD_FILE.xlsx
ReadFile for \\?\C:\Users\C22155\Documents\ID_PASSWORD_FILE.xlsx
ReadFile for \\?\C:\Users\C22155\Documents\ID_PASSWORD_FILE.xlsx
WriteFile for \\?\C:\Users\C22155\Documents\ID_PASSWORD_FILE.xlsx.Locked
DeleteFile for ../Documents/ID_PASSWORD_FILE.xlsx
CreateFile for WARNING_TEXT.txt
```

ユースケース(1/2)


メール
安全？
危険？
本当に
ダウンローダー？
ユーザー
実行ファイル
これはダウン
ローダーです
とりあえず実行しよう
ユーザー
実行ファイル
通常起動
怖いからランサムウェア
ワクチンを使おう
vaccine
DLL
ユーザー
実行ファイル
ランサムウェアワクチンを接種
してから起動

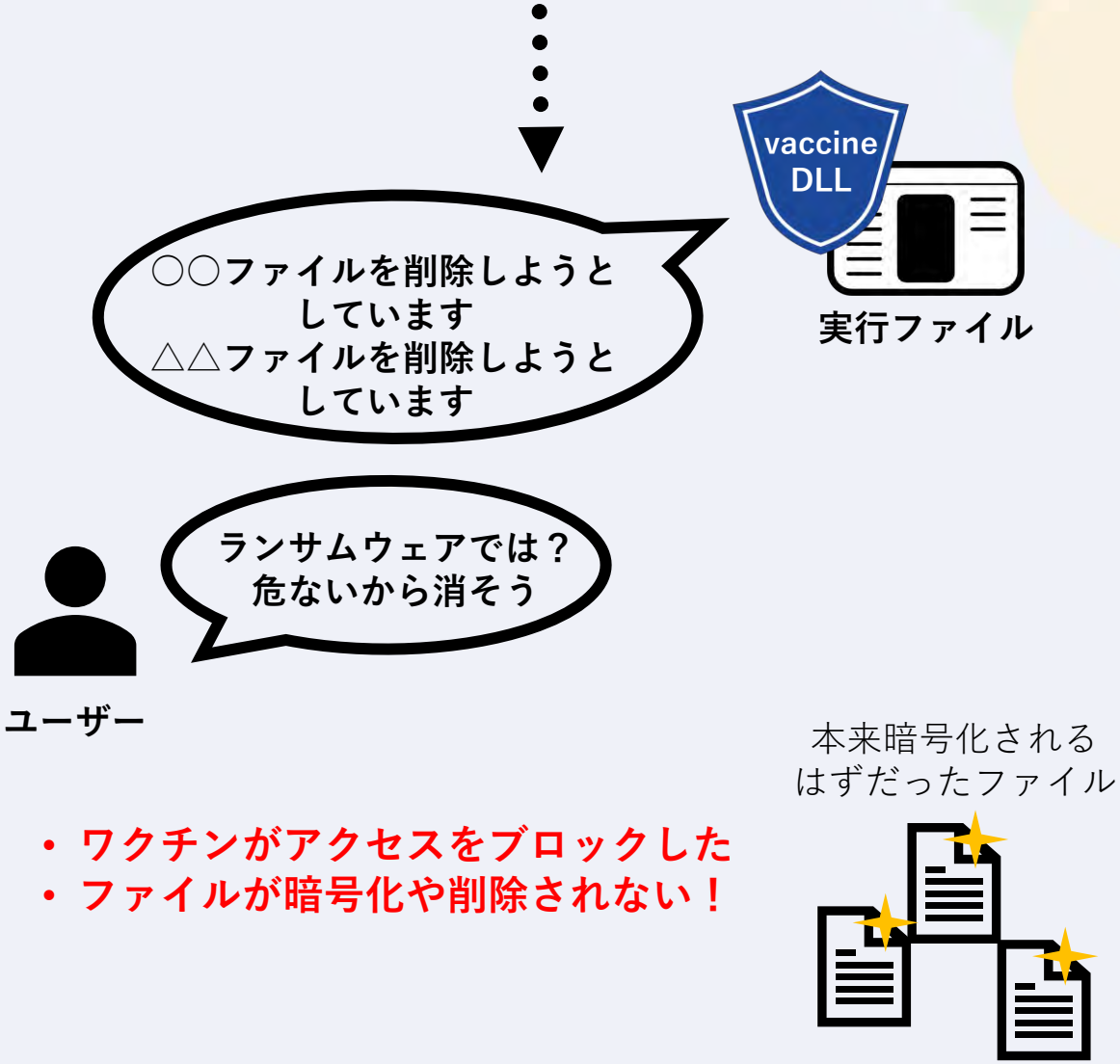
ユースケース(2/2)

通常起動



- ランサムウェアが起動してしまった
- ファイルが暗号化され使えなくなった

ランサムウェアワクチンを接種してから起動



- ワクチンがアクセスをブロックした
- ファイルが暗号化や削除されない！

評価

【防御できたか】

- ブロック成功:**3/7**件中
- ブロック失敗したものの影響
  - 接種成功:**2/4**件

前提条件
※ランサムウェアワクチンのリソースは**ランサムウェアのリソースに依存**します
※ランサムウェアが検閲orプロセスが削除されるまでの**最高値を値**としています
※全てのファイルアクセスをブロックするモードで行っています

【省リソース化できたか】

- ランサムウェアワクチンの方がWinDefより平均約**CPU50.0%, Memory337.8MB**⬇軽く防御することが出来た

仕様仮想環境:Virtual Box CPU:12th Gen Intel(R) Core(TM) i7-12700Hのプロセッサー6個, 使用率制限100% Memory:8192MB

ディフェンダー / ランサムウェア	ランサムウェアワクチン	Windows Defender
Akira	CPU:**11.4%**, Memory:**1.6MB**	CPU:**86.2%**, Memory:**319.8MB**
kawa	CPU:**0%**, Memory:**1.3MB**	CPU:**42.0%**, Memory:**323.7MB**
Nitrogen	CPU:**1.9%**, Memory:**1.3MB**	CPU:**35.2%**, Memory:**374.1MB**

ランサムウェアワクチンの適用範囲

- ランサムウェアワクチンは**特定の実行ファイルから直接呼ばれたファイルアクセスを軽量に監視**する

【防御範囲】

- ファイルパスが存在するランサムウェアからのアクション

【防御範囲外】

- ファイルパスが存在しないランサムウェアからのファイルアクセス
  例:ファイルレスマルウェア(メモリ内のみで動くマルウェア)

- ターミナル等のコマンド、低レイヤーなアセンブリで関数を
  実行してくるランサムウェア
  例:LOL攻撃, shellcodeやsyscallを使った関数呼び出し

- ファイルアクセスを行わずに攻撃を行うマルウェア
  例:ランサムウェア以外のマルウェア

社会実装について

修了生3名, 元トレーナーにレビュー頂いた

【防御できない技術・手法】

- shell codeやsyscall
- ターミナルコマンドを用いた攻撃
- レジストリの改変
- メモリ書き込み
- 既存プロセスへのInjection

↓解決策

- ターミナル書き込み関数のHook
- レジストリ変更の関数をHook
- メモリ書き込みの関数のHook
- 既存プロセスへのアタッチ関数のHook

【新たな活用法】

- モニター,デバッガーの+α
  - 関数呼び出しのモニター
  - ブレイクポイント設置機能の追加
  - 高レイヤーなモニター
- 仮想環境の+α
  - Hookする関数の指定
    - 用途ごとに関数をまとめる
- 通知・ログの方式を追加

防御範囲が狭い, 漏れがある可能性がある
→ないよりはあるほうがいいもの

まとめ

- **背景**
  今までののディフェンダーは**リソース消費量が多い**
  またランサムウェアは必ず**ファイルアクセス**を行う

- **提案手法:ランサムウェアワクチン**
  ファイルアクセスを**IAT Hook**を用いて省リソースで動的な防御手法

- **評価**
  ランサムウェアの **3/7 ブロックが出来た**
  どのファイルにどんなアクセスをしているかの確認をすることが出来た
  従来のソフトより約 **CPU50.0%, Memory337.8MB省リソース**
  でブロックが出来た

- **今後**
  オプションの追加、アンチウイルスとしての機能以外の改善、接種手法の改善