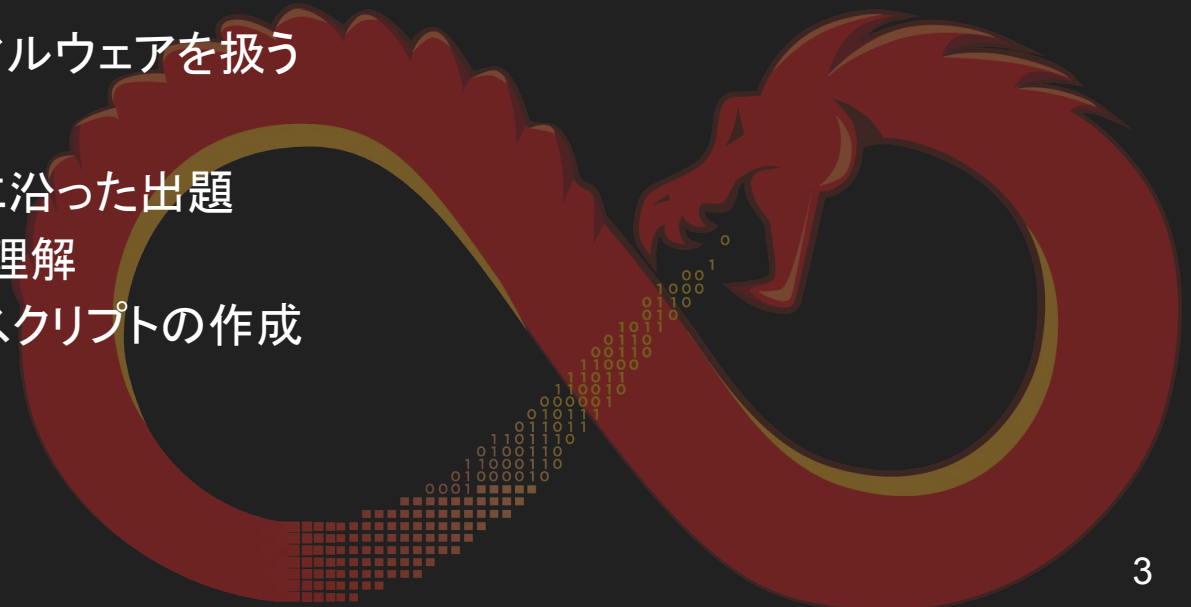
MWS Cup 2024
静的解析 振り返り

2024の問題担当

- Static Analysis主担当
  - 中島 将太 (株式会社サイバーディフェンス研究所)
- 問題作成委員
  - 桑原 翼 (株式会社FFRIセキュリティ)
  - 皆川 諒 (株式会社エヌ・エフ・ラボラトリーズ)
  - 末廣 繁樹 (株式会社エヌ・エフ・ラボラトリーズ)
  - 光安 正憲 (西日本電信電話株式会社)
  - 清水 嶺 (西日本電信電話株式会社)
  - 緒方 湧己 (西日本電信電話株式会社)
  - 仲川 宜秀 (西日本電信電話株式会社)

テーマ

- マルウェアを正しく理解する
  - 課題を通して解析のポイントを学習する
- 最新情報を得る
  - 最近のin-the-wildなマルウェアを扱う
- 実務に近い作業
  - マルウェアのトレンドに沿った出題
  - マルウェアのコードの理解
  - 静的解析による復号スクリプトの作成

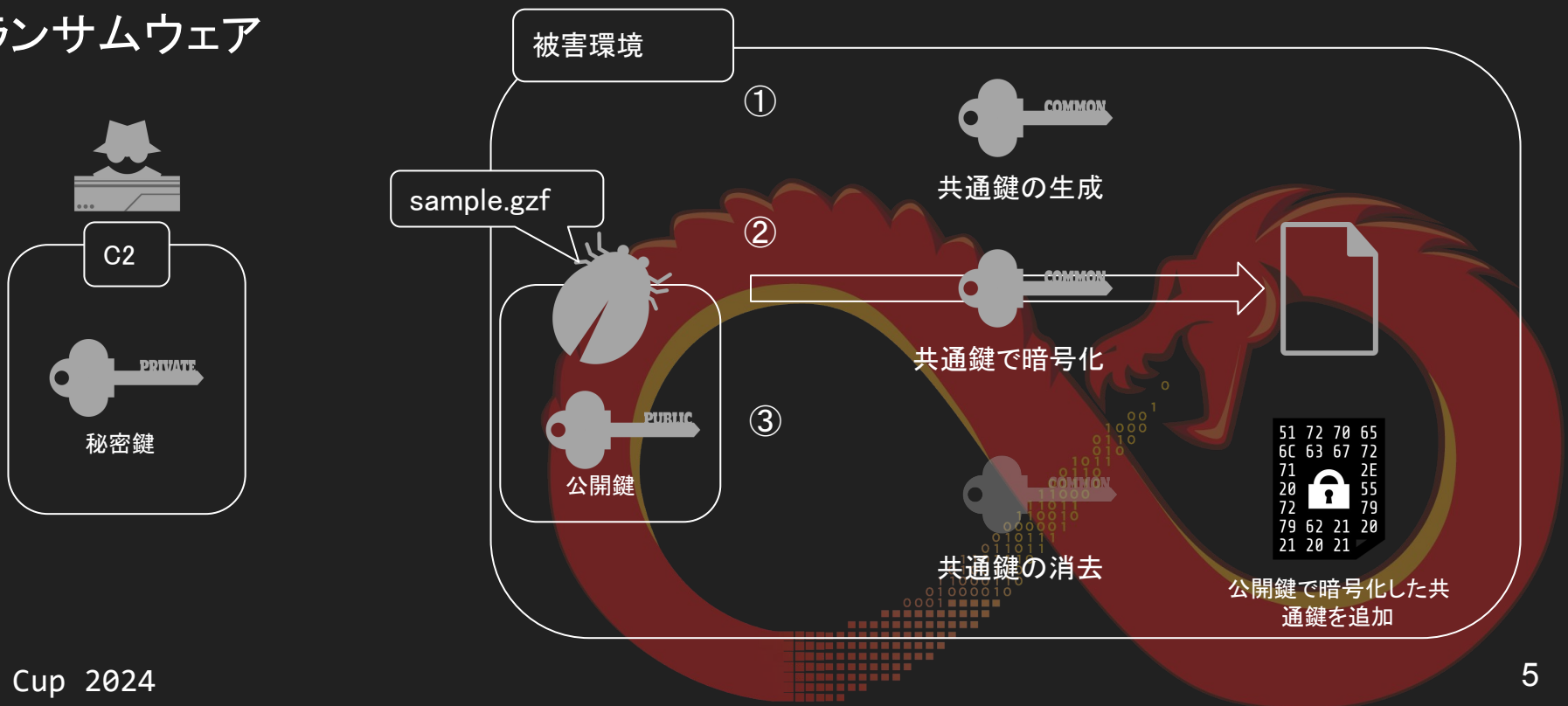


ポイント

- 積極的に変数名や型を変更する
  - 名前を付けて読みやすくしていく
  - デフォルトでは型情報がないことが多い
- デコンパイラを信用しすぎない
  - アセンブリを確認して整合性を確認する
  - 手動で修正する
- 順番に回答する必要はないので解けそうな問題から解く

マルウェアの動作概要

ランサムウェア



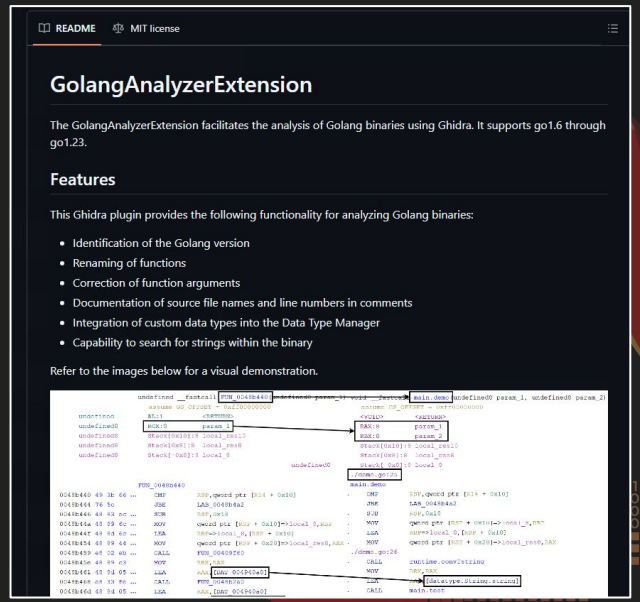
加工済みのGZF

- MWSの問題作成にあたって以下の変更を加えています
  - GolangAnalyzerExtensionのAnalyze結果の適用
  - 出題に伴う関数名と関連するメタ情報の修正
  - コード中で参照される画像データの差し替え
  - 補助のための構造体の適用

今年は初めてGo言語のマルウェアを出題しました
二週間ほど前に出題することをアナウンスして
ヒントも共有しました

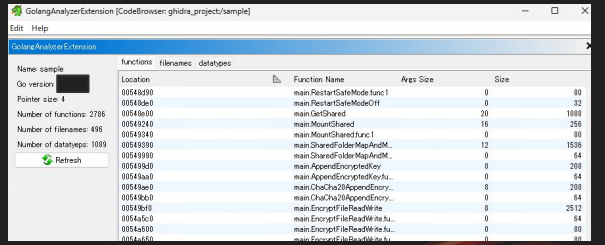
GolangAnalyzerExtension

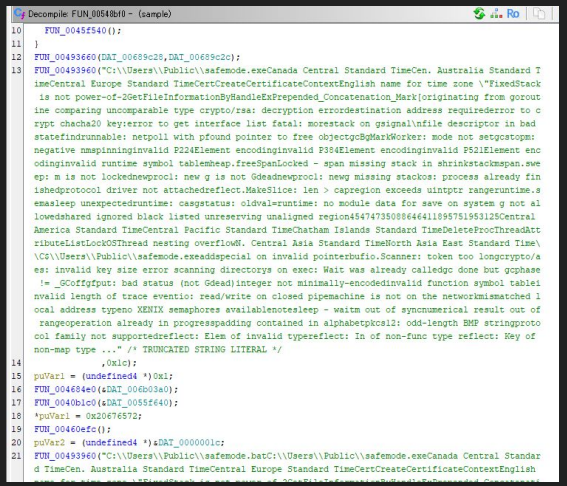
- 問題作成委員FFRIの桑原さんが開発したGo言語バイナリ解析用のGhidra Extension

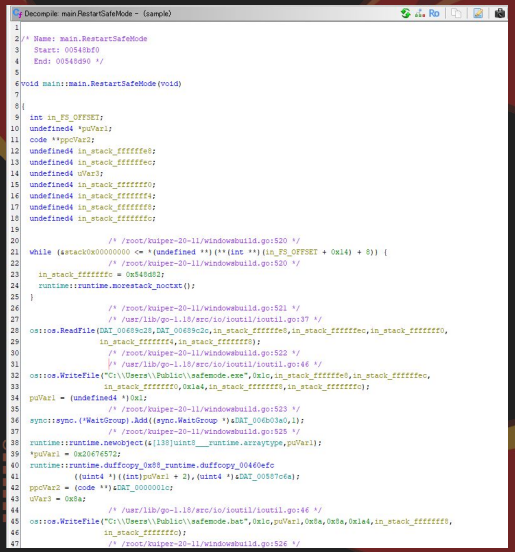


GitHub - mooncat-greenpy/Ghidra_GolangAnalyzerExtension: Analyze Golang with Ghidra

GolangAnalyzerExtension

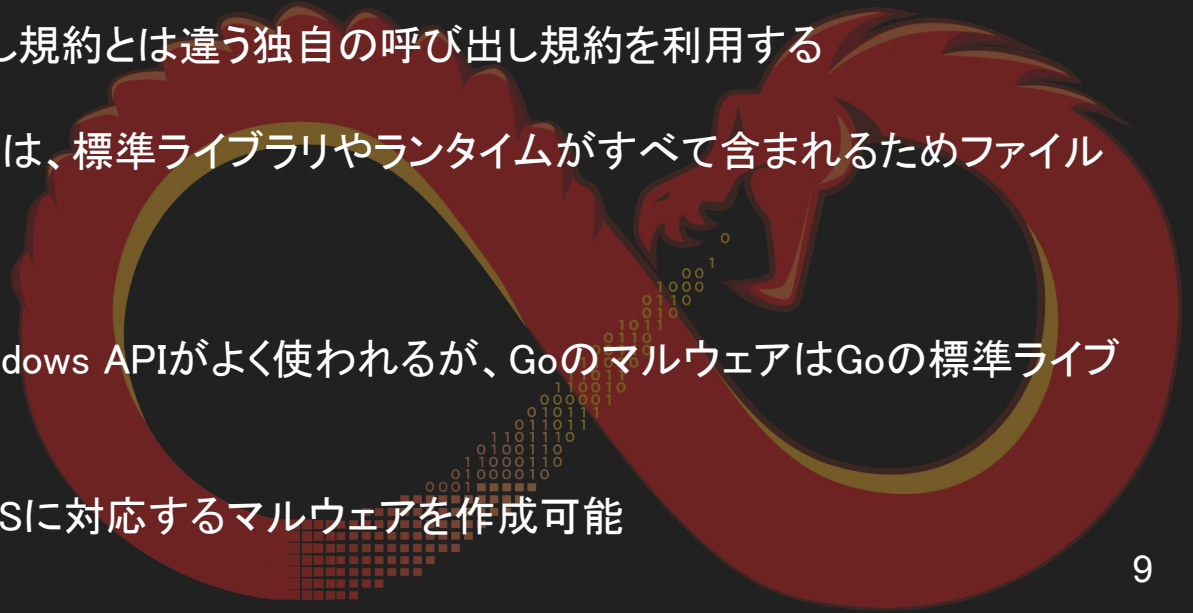






ヒント: Go言語のマルウェアの特徴

- 特有のデータ構造
  - Go固有の文字列フォーマットや関数名の関数情報の構造を持つ
  - ツールで対応可能
- 独自の呼び出し規約
  - C++で利用される呼び出し規約とは違う独自の呼び出し規約を利用する
- スタティックリンク
  - Go言語の実行ファイルには、標準ライブラリやランタイムがすべて含まれるためファイルサイズが大きい
  - 必要なコードだけを読む
- 標準ライブラリの使用
  - C++のマルウェアではWindows APIがよく使われるが、Goのマルウェアは Goの標準ライブラリを利用する
- クロスプラットフォーム対応
  - 1つのコードから複数のOSに対応するマルウェアを作成可能



ヒント: Go言語特有のデータ構造

Go言語特有のデータ構造をいくつか紹介する

- 文字列
  - 文字列へのポインタ、文字列長で構成される

```
struct string {
    char* str;
    int len;
}
```

Go言語で以下のように記載されている

```
var testStr string = "test"

func testString(s string) *string {
...
}
```

- スライス（可変長の配列のようなもの）
  - 配列へのポインタ、配列の長さ、メモリ確保された配列の長さ

```
struct slice {
    void* array;
    int len;
    int cap;
}
```

Go言語で以下の第1引数のように記載されている

```
func testSlice(s []int) {
…
}
```

ヒント: Go言語独自の呼び出し規約

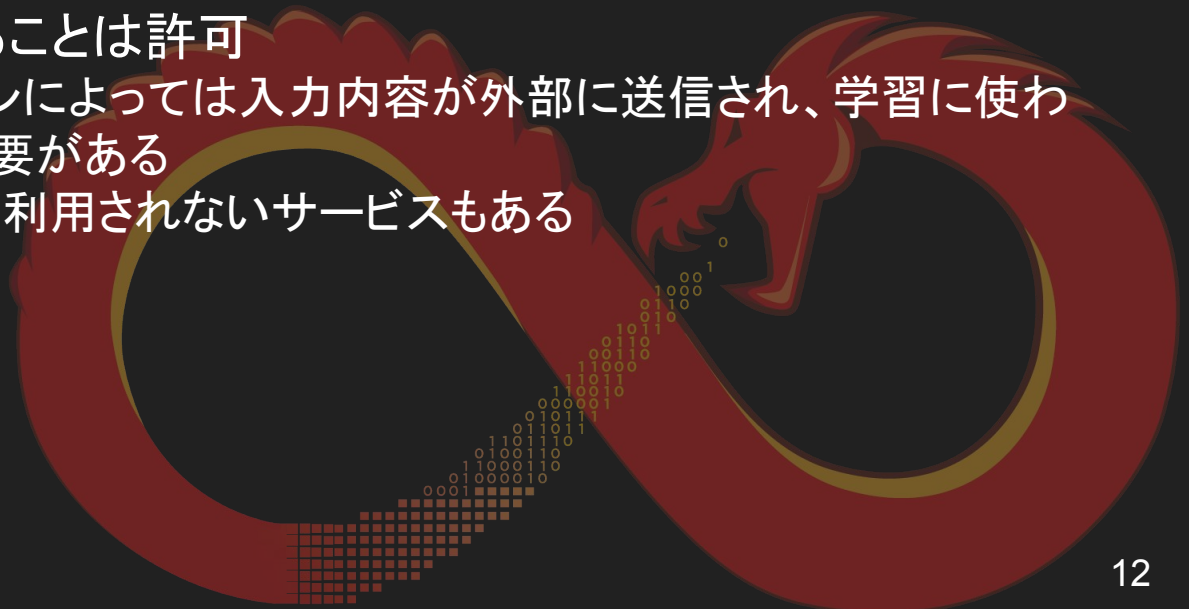
Go言語独自の呼び出し規約について紹介する

- 以下の図に関数呼び出し時の引数・戻り値が格納される場所を載せる
  - Go言語のバージョンやアーキテクチャにより変化
  - 例外もあるため注意すること

	64bit	32bit
Go 1.17以上	引数: rax, rbx, rcx, rdi, rsi, r8, r9, r10, r11, スタックの先頭から 戻り値: 引数と同じ（raxから利用）	左下と同じ
Go 1.17より前	引数: スタックの先頭から 戻り値: 引数と同じ（引数で利用してないもののみ利用）	左と同じ

ヒント: GoogleやAIの活用

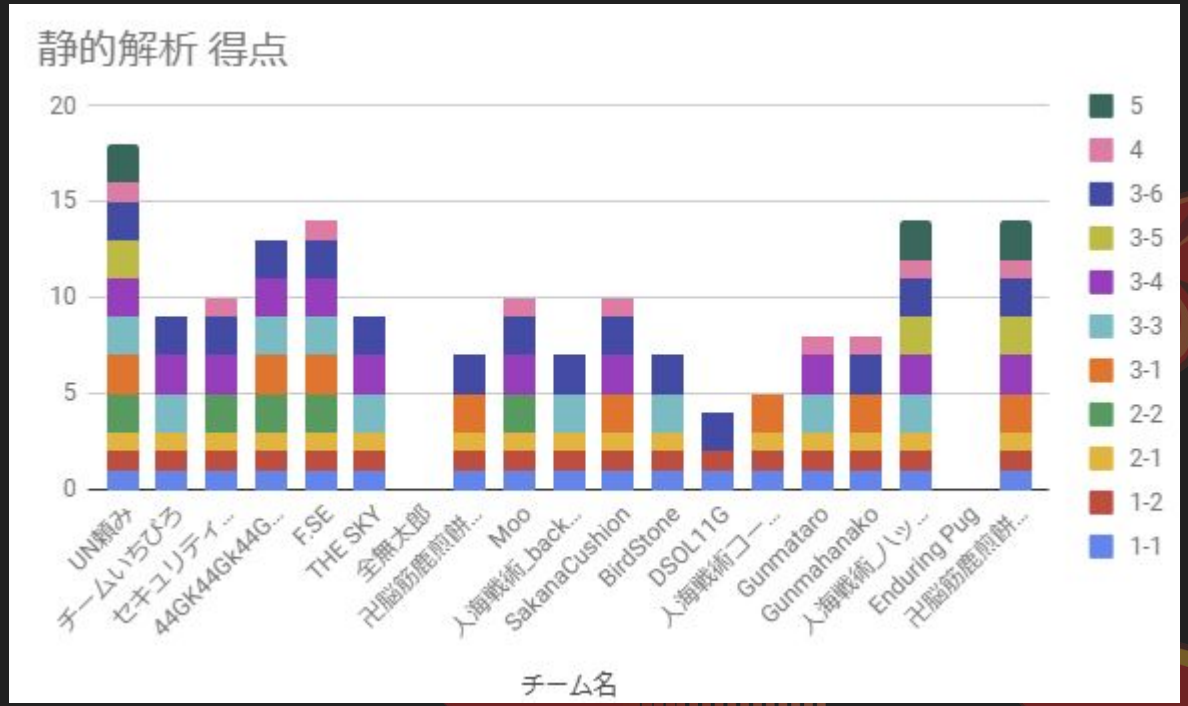
- ファイルを直接VirusTotalなどの外部サービスにアップロードすることは禁止
- ただし、コードの断片や文字列をGoogle検索、Chat GPTなどのGenerative AIのサービスに投稿することは許可
  - 実務で扱う際は、プランによっては入力内容が外部に送信され、学習に使われることを考慮する必要がある
  - 有料プランでは学習に利用されないサービスもある



問題一覧

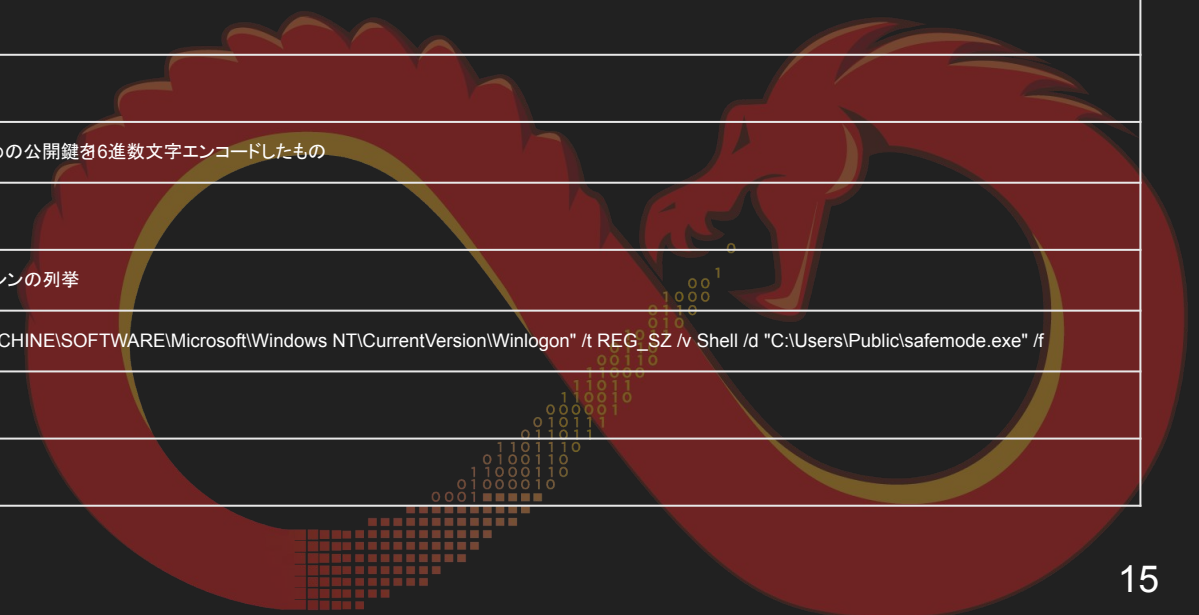
1-1. Goメタ情報 1	1-2. Goメタ情報 1	2-1. オプション 1	4. 画像抽出 1
2-2. オプション 2	3-1. 関数読解 2	3-2. 関数読解 2	3-3. 関数読解 2
3-4. 関数読解 2	3-5. 関数読解 2	3-6. 関数読解 2	5. 暗号鍵 2
6-1. データ復号 2	6-2. データ復号 3		

結果


静的解析 得点
20
15
10
5
0
UN頼み
チームいちびろ
セキュリティ…
44GK44Gk44G…
F.SE
THE SKY
全無太郎
卍脳筋鹿煎餅…
Moo
人海戦術_back…
SakanaCushion
BirdStone
DSOL11G
人海戦術コー…
Gunmataro
Gunmahanako
人海戦術_ハッ…
Enduring Pug
卍脳筋鹿煎餅…
チーム名
5
4
3-6
3-5
3-4
3-3
3-1
2-2
2-1
1-2
1-1

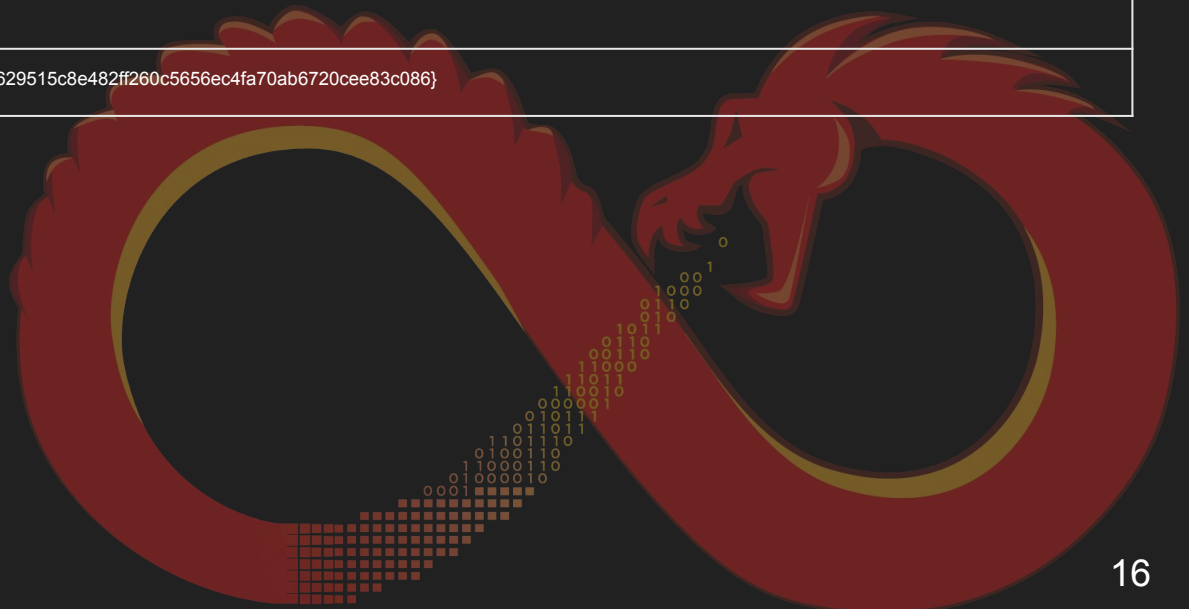
解答まとめ

1-1. Goメタ情報	go1.18.1
1-2. Goメタ情報	005511b0
2-1. オプション	flag
2-2. オプション	0066e1c0
3-1. 関数読解	暗号化鍵、IVを暗号化するための公開鍵を6進数文字エンコードしたもの
3-2. 関数読解	b,c,d
3-3. 関数読解	Local Network に存在するマシンの列挙
3-4. 関数読解	reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon" /t REG_SZ /v Shell /d "C:\Users\Public\safemode.exe" /f
3-5. 関数読解	596f7572
3-6. 関数読解	デスクトップの壁紙を変更する

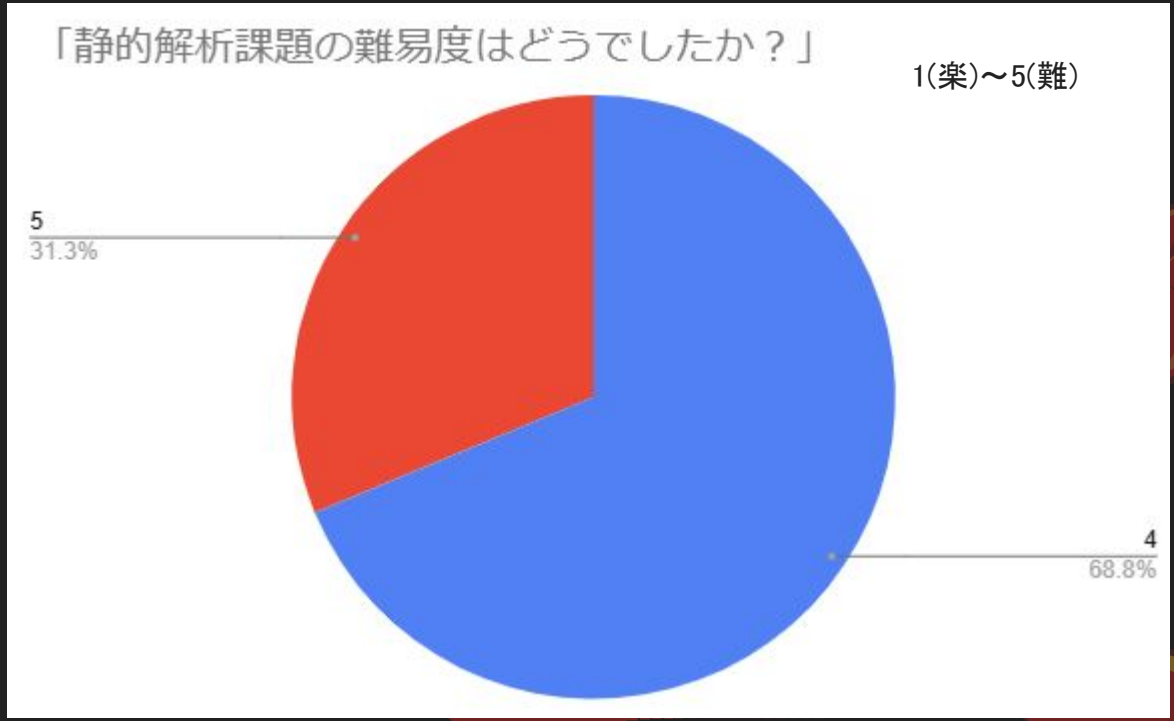


解答まとめ

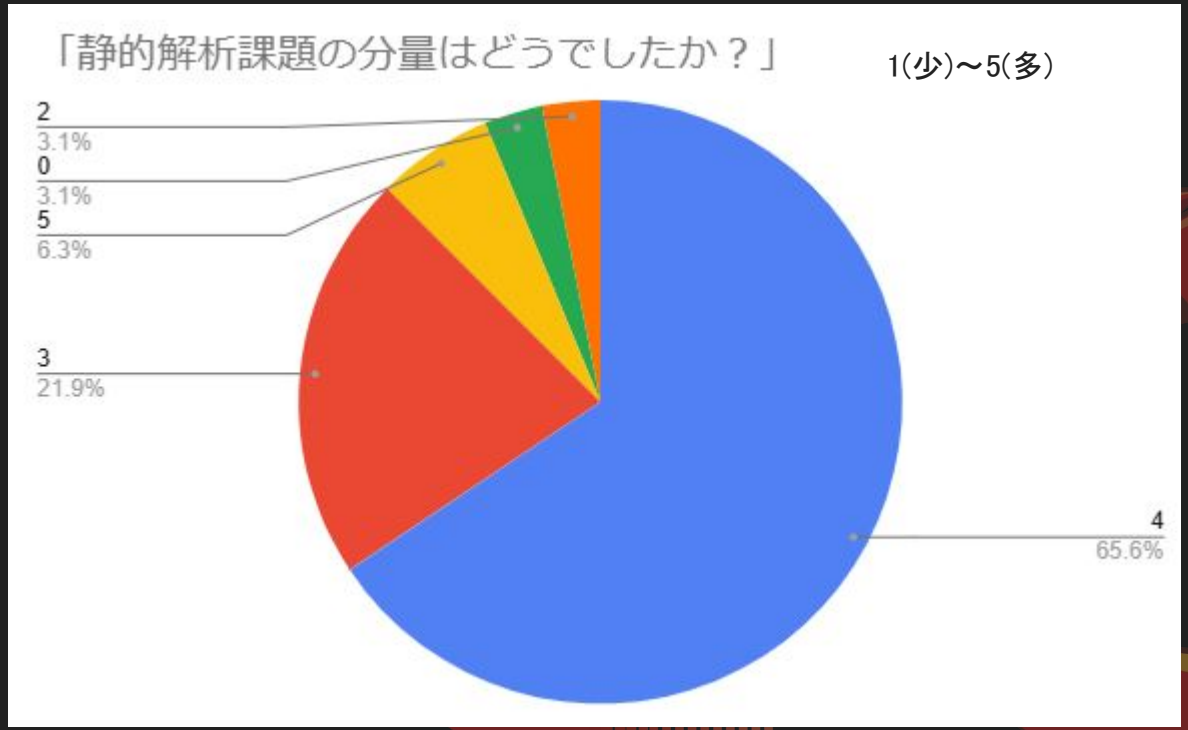
4. 画像抽出	MWS{want_the_flag_pay_up}
5. 暗号鍵	0054d8d0
6-1. データ復号	8fc7488d
6-2. データ復号	MWS{fb9acf8aea87f456d4ec629515c8e482ff260c5656ec4fa70ab6720cee83c086}



アンケート結果



アンケート結果

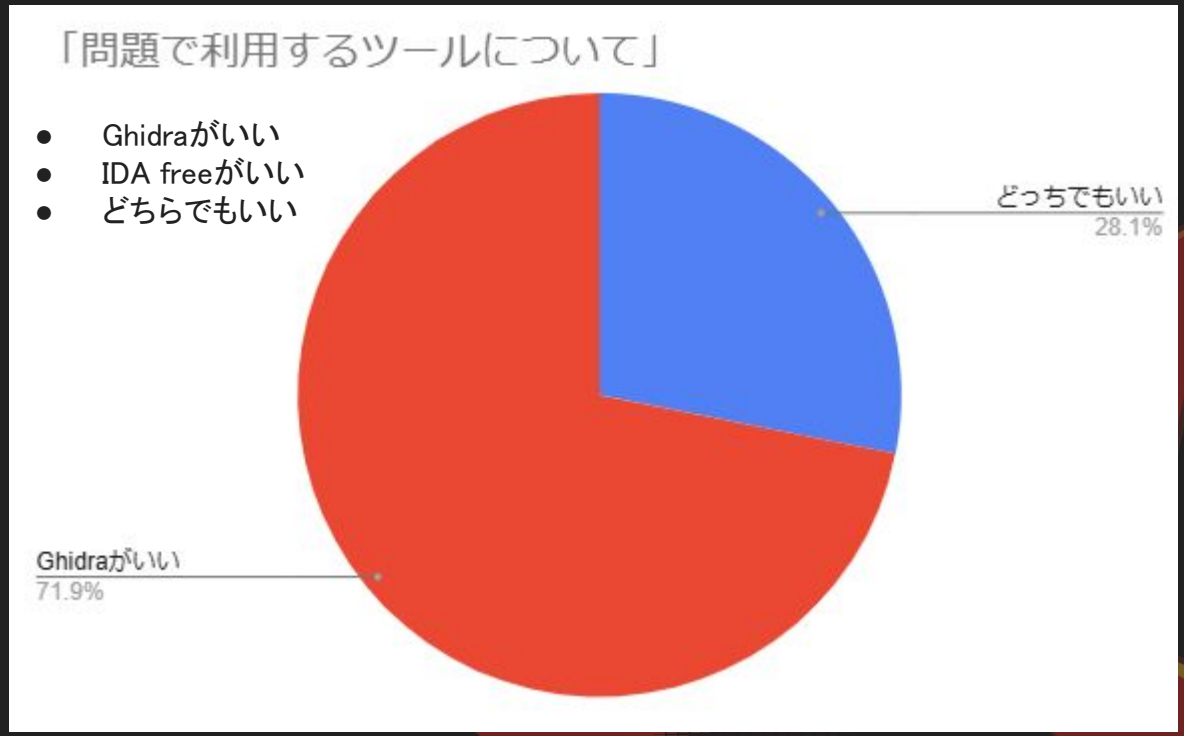


アンケート結果

課題の解答にLLM(Chat GPTなど)を活用しましたか？

- した 27/30
- していない 2/30
- 無回答 1/30

アンケート結果


「問題で利用するツールについて」
Ghidraがいい
IDA freeがいい
どちらでもいい
どっちでもいい
28.1%
Ghidraがいい
71.9%

アンケート結果

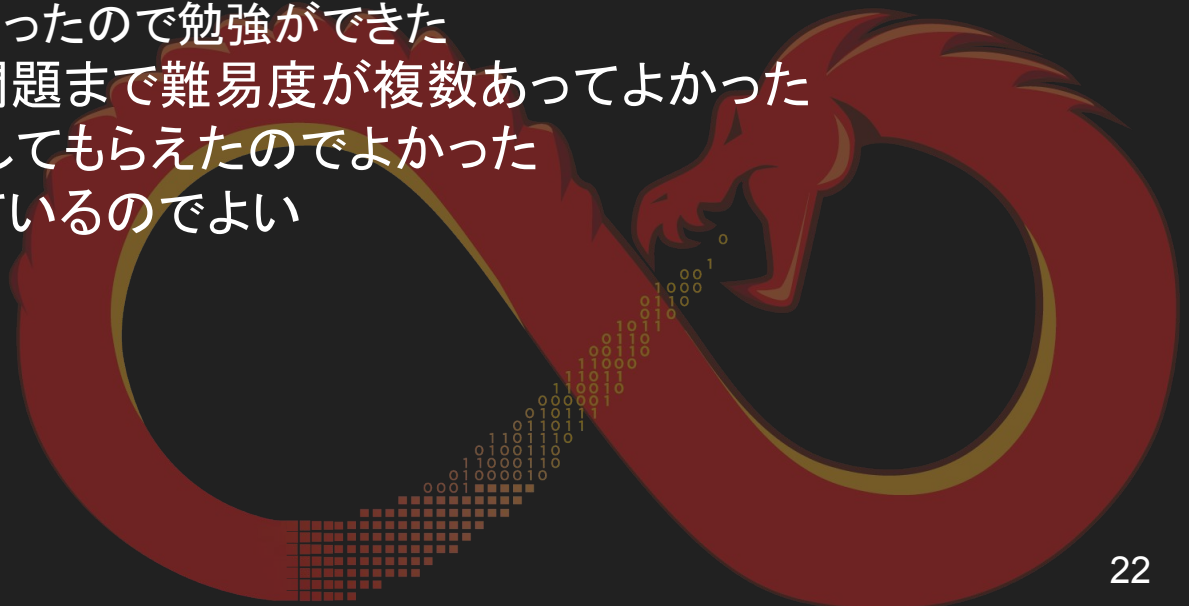
静的解析として取り上げてほしい内容があれば教えてください。

- Rust製マルウェア
- RATが多いので他の種類のマルウェアもやりたい
- 耐解析機能や難読化の解除について
- アプリケーションの脆弱性に攻撃するマルウェア
- Androidマルウェア
- アンチディスアセンブリ
- 作問陣が見た最近のトレンドを含んだ興味深いマルウェア
- ワイパー

アンケート結果

今回の 静的解析課題に関して良かった点

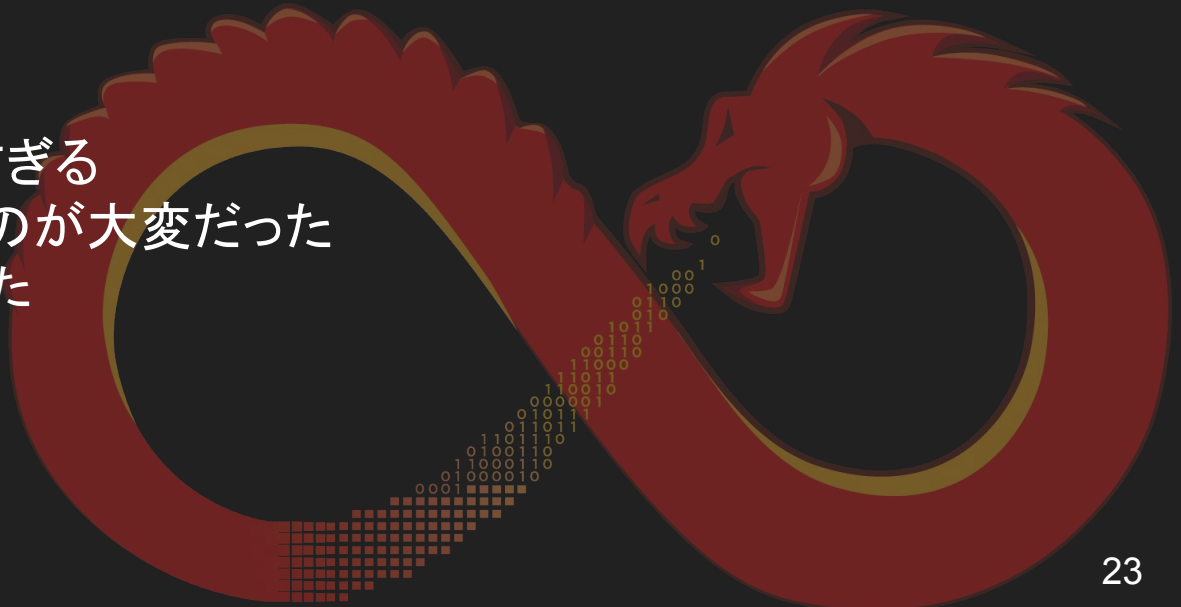
- Go言語のマルウェアに関する出題が好評だった
  - 新しい技術に触れることができた
  - 事前にアナウンスがあったので勉強ができた
- 簡単な問題から難しい問題まで難易度が複数あってよかった
- 事前に参考資料も共有してもらえたのでよかった
- Ghidraの資料が充実しているのでよい



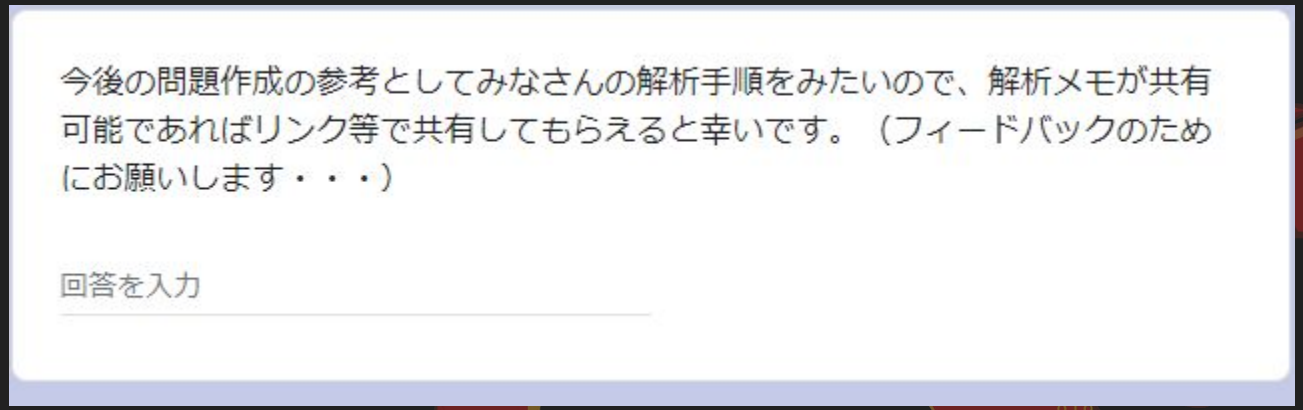
アンケート結果

今回の静的解析課題に関して悪かった点、難しかった点

- もうちょっとヒントが欲しい
- ツールの使い方を説明してほしい
- 難しすぎた
- 初心者にはきつい
- LLMで解ける問題が多すぎる
- Go言語の仕様に慣れるのが大変だった
    - だけどいい勉強になった



ご協力ありがとうございました



来年度の参考にさせていただきます！