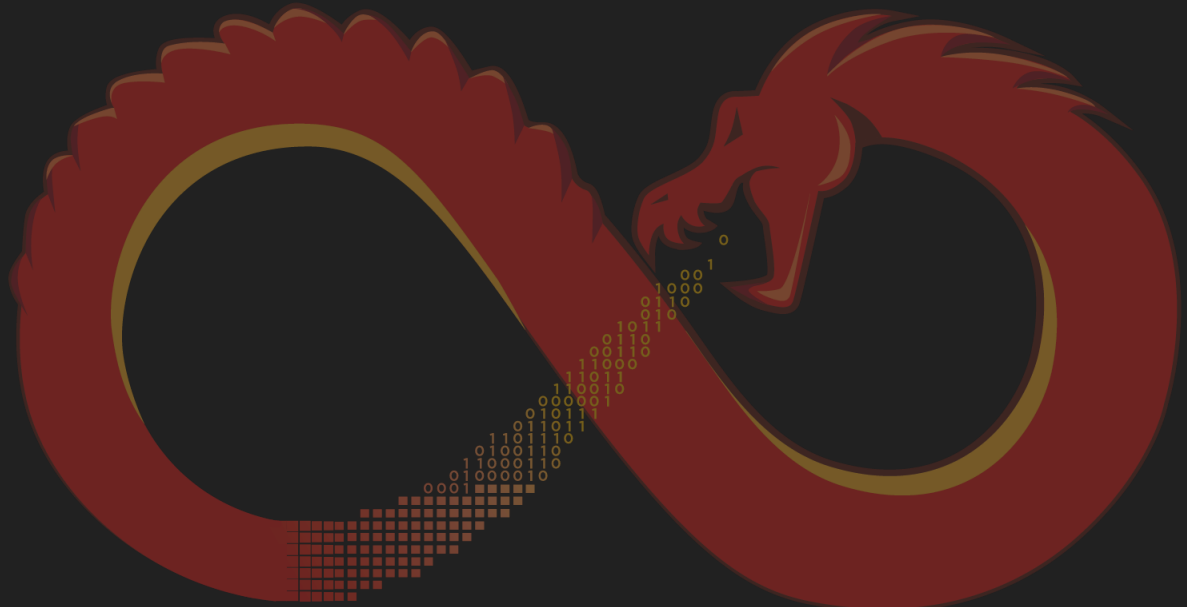
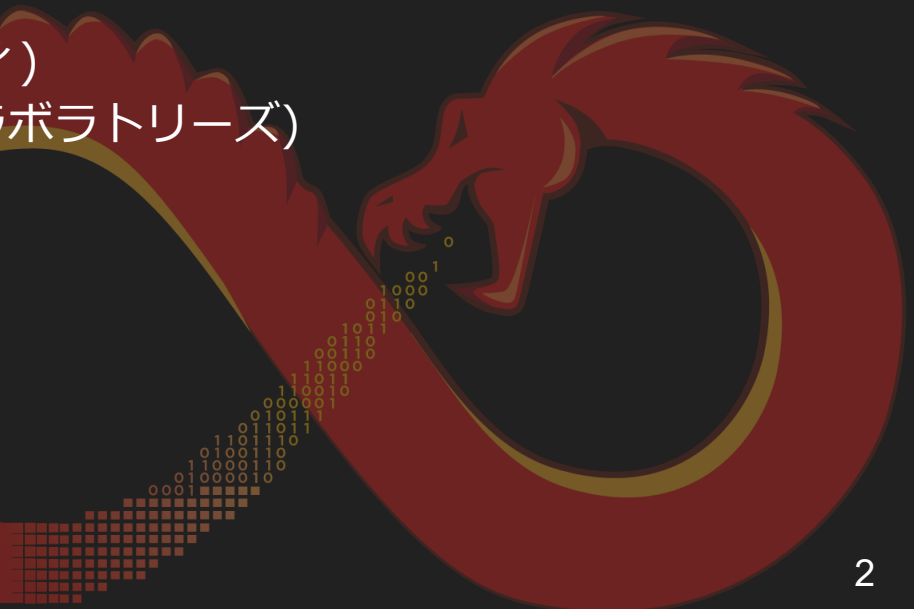
MWS Cup 2025
静的解析 振り返り



2025の問題担当

- 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)
  - ○ 末廣 繁樹 (株式会社エヌ・エフ・ラボラトリーズ)
  - ○ 山田 裕彌 (株式会社ラック)
  - ○ 川越 謙宏 (工学院大学)
  - ○ 仲川 宜秀 (NTT西日本株式会社)
  - ○ 二瓶 凌輔 (NTT西日本株式会社)
  - ○ 緒方 湧己 (NTT西日本株式会社)
  - ○ 根津 泰之 (NTT西日本株式会社)



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
  - 最新の技術を活用して効率よく解析する(LLMの利用など)

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
- LLMを信用しすぎない
  - 自分で正誤を確認できる技術を身につける
- 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

Ghidra Tips: 構造体の定義

- 構造体の定義の流れ
  - ディスアセンブル、デコンパイルから構造体と思われるコードを見つける
  - コードを解析して、構造体のメンバーを解析
  - ヘッダファイルを作成
  - Ghidraで定義したヘッダファイルを読み込み
  - デコンパイル画面で構造体を適用する

```
printf("Header: magic=0x%08X count=%u\n",(ulonglong)*_DstBuf,
       (ulonglong)(ushort)_DstBuf[1]);
printf("C2: beacon_ms=%u jitter=%u retry_max=%u\n",(ulonglong)_DstBuf[2],
       (ulonglong)(byte)_DstBuf[3],(ulonglong)*(byte *)((longlong)_DstBuf + 0xd));
printf("C2: campaign_id=\'%s\'\n",_DstBuf + 4);
printf("C2: user_agent=\'%s\'\n",_DstBuf + 8);
if ((short)_DstBuf[1] != 0) {
```

```
printf("Header: magic=0x%08X count=%u\n",(ulonglong)conf->magic,(ulonglong)conf->count);
printf("C2: beacon_ms=%u jitter=%u retry_max=%u\n",(ulonglong)conf->beacon_ms,
       (ulonglong)conf->jitter_percent,(ulonglong)conf->retry_max);
printf("C2: campaign_id=\'%s\'\n",conf->campaign_id);
printf("C2: user_agent=\'%s\'\n",conf->user_agent);
if (conf->count != 0) {
```

ヒント: インターネットと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
  - 学習に利用されない設定もある
- マルウェアの中にあるIPアドレス/URL/ドメインに直接アクセスすることは禁止。下記の手段で確認すること。
  - Googleなどの検索エンジンによる検索
  - urlscan（https://urlscan.io/）の利用

問題一覧

ファイル配布 0	1-2. 文字列の難読化 2	1-3. APIの構造体 2	1-4. コマンド1 2
1-5. コマンド2 2	2-1. Config読解 2	2-2. Config読解 2	3-1. 通信データの符号化・暗号化 2
4-1. 文字列加工 2	1-1. 動的APIアドレス解決 3	3-2. 通信データの符号化・暗号化 3	4-2. ファミリ名 3

問題ジャンル

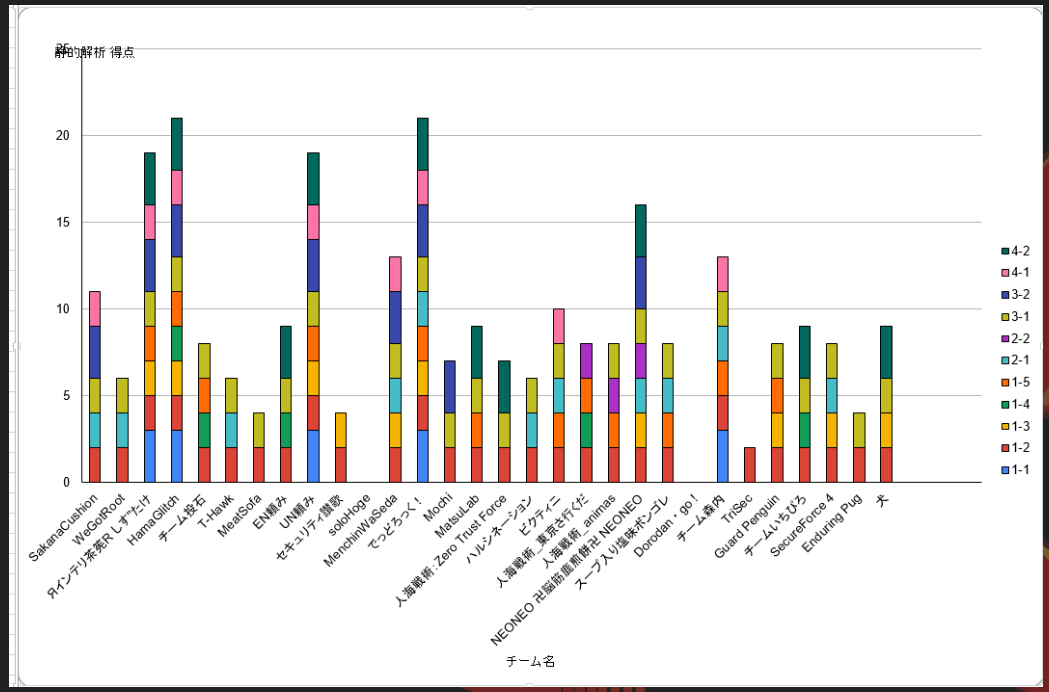
- API難読化解除
  - 1-1
  - 1-2
  - 1-3
- コマンド解析
  - 1-4
  - 1-5
- コンフィグ解析
  - 2-1
  - 2-2
- 復号系
  - 3-1
  - 3-2
- アトリビューション
  - 4-1
  - 4-2

解答まとめ

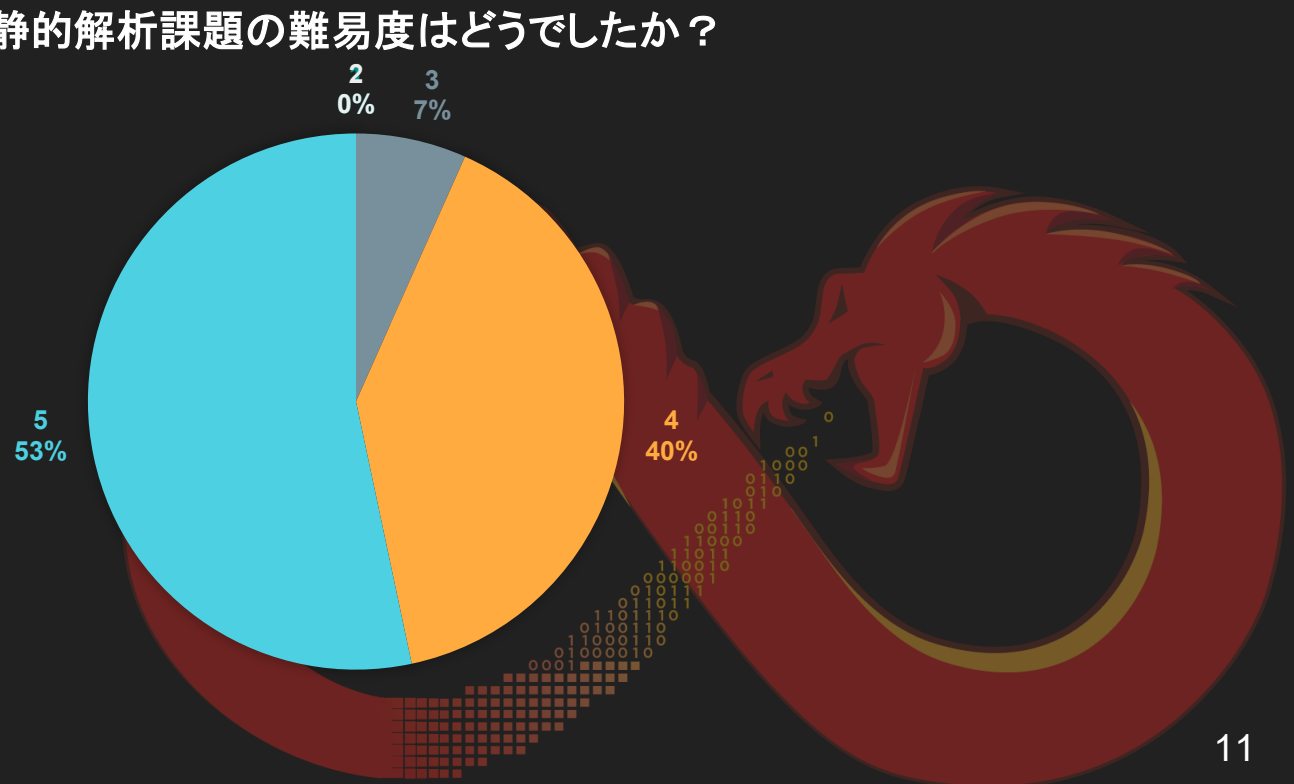
1-1. 動的APIアドレス解決	エ,シ,テ,オ,ニ,イ,キ,フ,カ,ノ,チ,ナ,ク,コ
1-2. 文字列の難読化	4P7_U53_34SY_3NC0D3
1-3. APIの構造体	CreateProcessW,ReadFile,Sleep
1-4. コマンド1	17
1-5. コマンド2	18000edd8
2-1. Config読解1	f. クライアントのユーザ名
2-2. Config読解2	g. クライアントのグローバルIPアドレス
3-1. 通信データの符号化・暗号化	a. RC4をカスタマイズしたアルゴリズム
3-2. 通信データの符号化・暗号化	1d3n71fy1n9_cu57om_rc4_15_4n_3553n71a1_5k1ll
4-1. 文字列加工	b_905QD4656:H
4-2. ファミリ名	APT-C-60_SpyGlace

結果

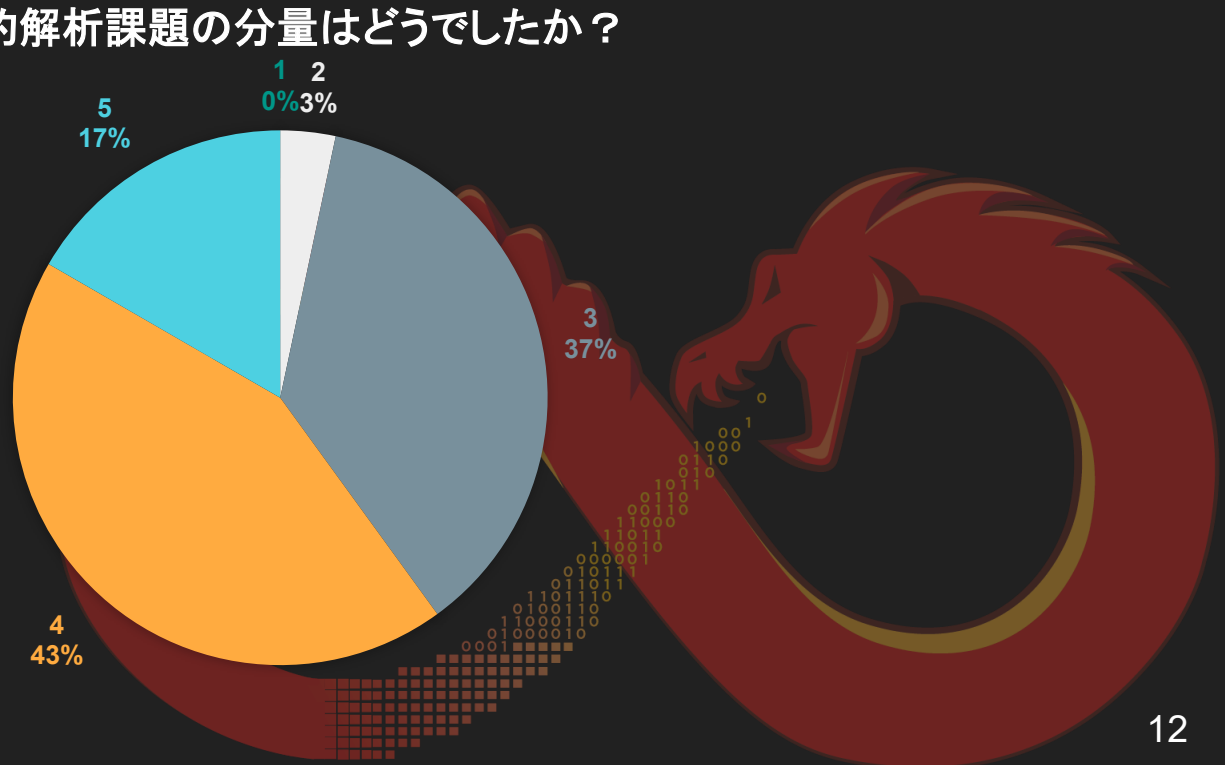
- 全問題について、どこか1チーム以上は解いた形



アンケート結果

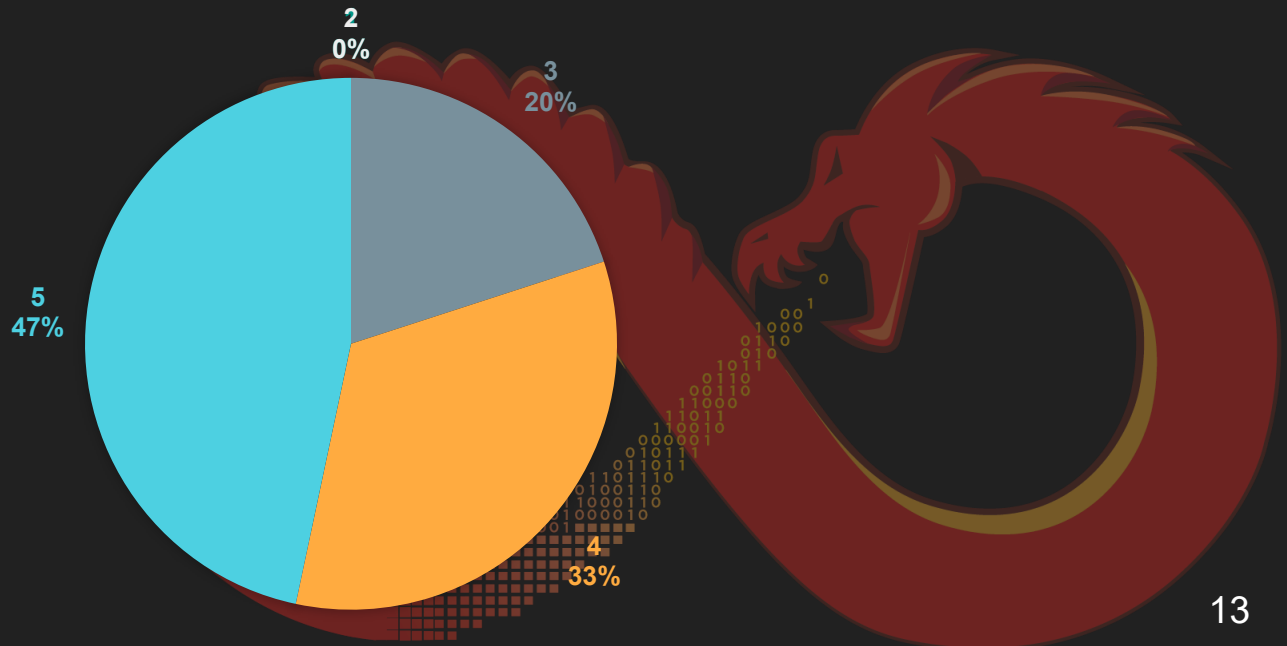


アンケート結果



アンケート結果

過去問に比べて課題自体の難易度はどう変化したと思いますか？



アンケート結果

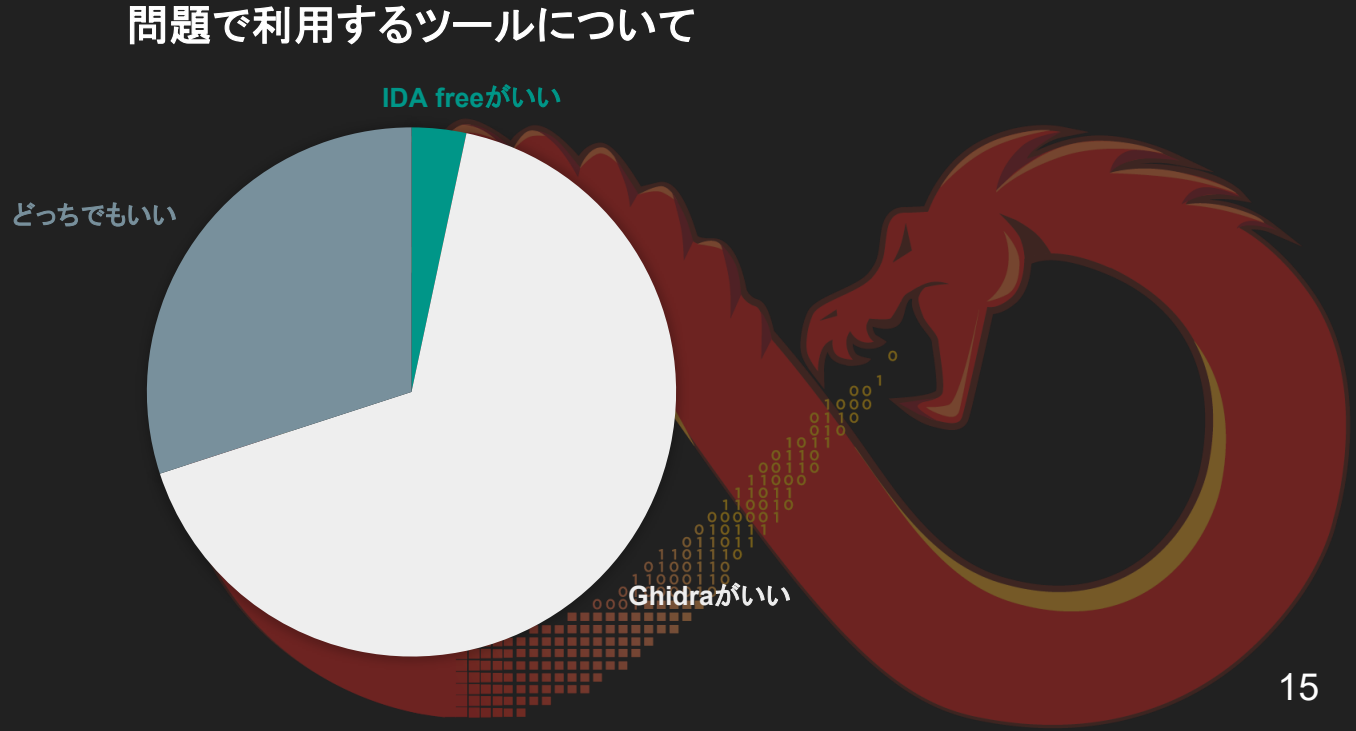
課題の解答にGhidraMCPを活用しましたか？

Yes　19名　No　11名

利用したLLMのモデルは何ですか？

	人数（重複あり）
ChatGPT系	10
Claude系	9
Gemini系	7
Grok	1
Copilot	1

アンケート結果



アンケート結果

生成AIについては賛否両論あり。（今後の作問の方針について要検討）

・生成AIを使用することにより、難易度が上がったため、回答のハードルが上がったと感じたと

・生成AI環境を使用するにあたり、無料版だとどうしても制限がかかってしまうため、不便に感じた

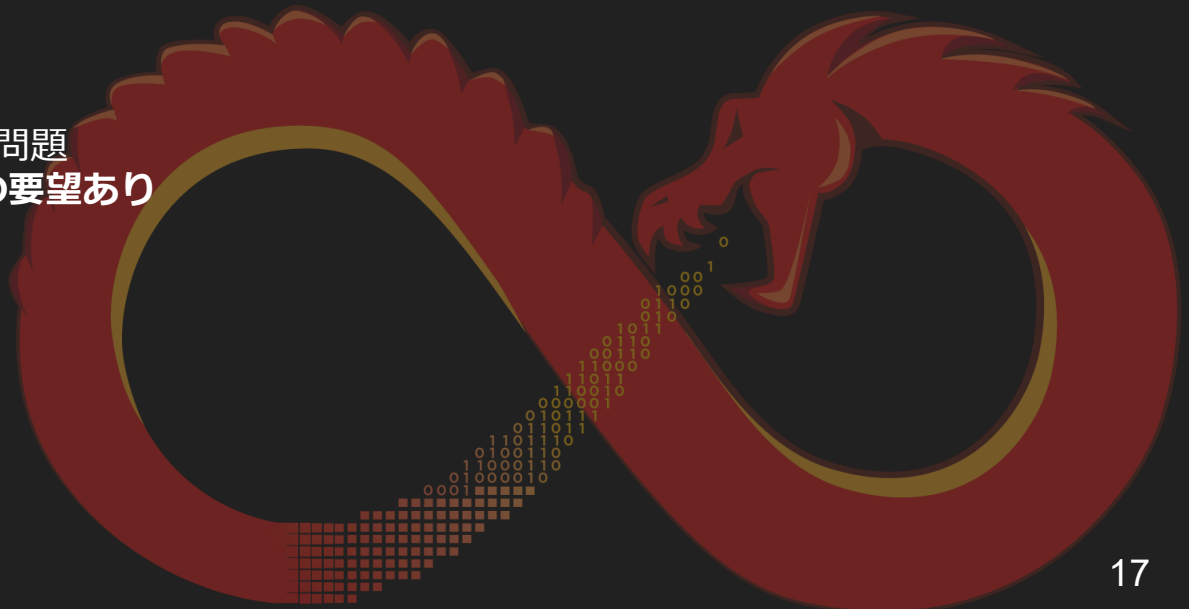
・AIを使うことでだけでは答えが出ず、自身の判断で解析を行っていくプロセスが大事だと痛感した、またそれをもとに自分で変数や関数を追いかけるなど勉強になったとポジティブなコメントもあり

・ある程度のレベルまで学習していないと、ただLLMを使うだけになってしまった

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。

- 暗号解読
- 難読化
- テイント解析
- rat,ランサムウェア以外
- レジスタ・アセンブリを意識した問題
- **Rustマルウェアについては多くの要望あり**



アンケート結果

皆様、アンケートへのご回答ありがとうございました！