Rustマルウェア解析

学習駆動社会実装ゼミ
13Sx 川越謙宏

一言で

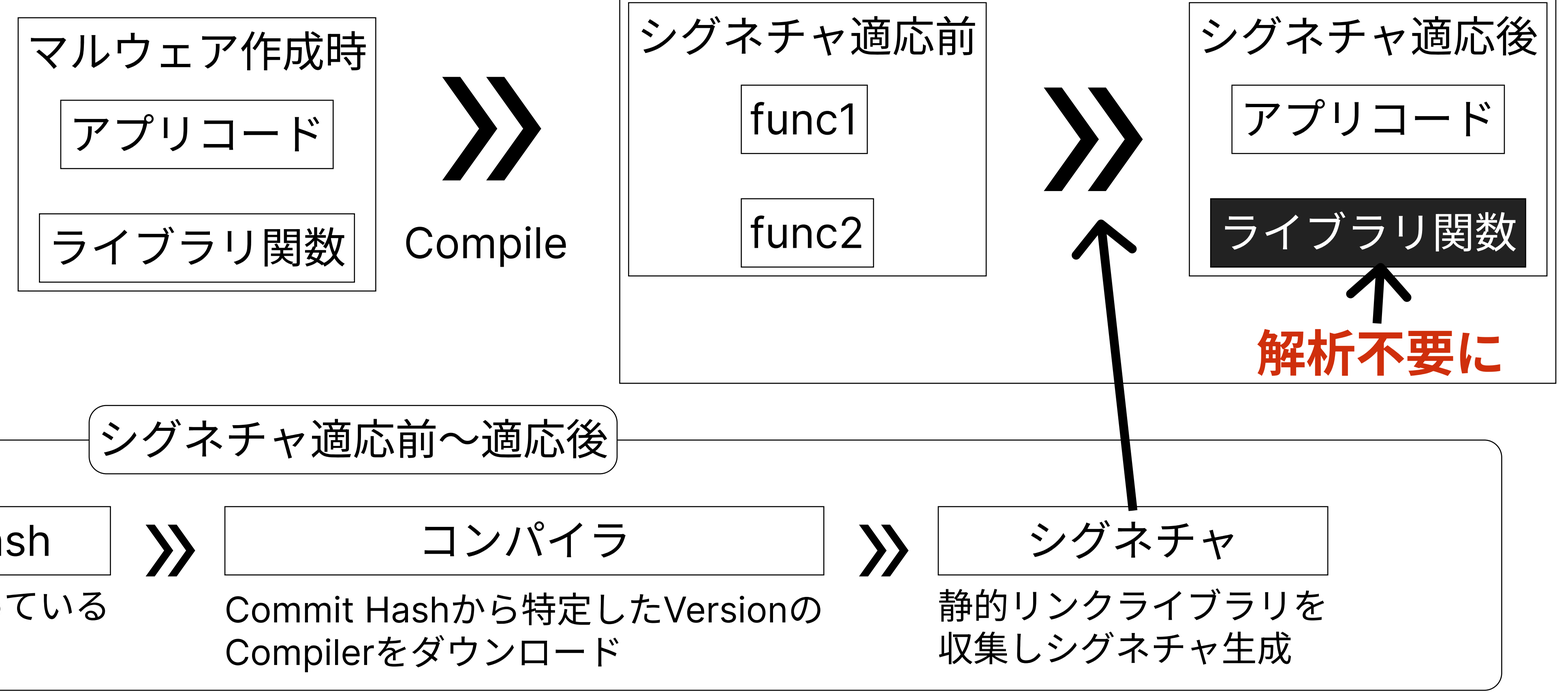
OSSツールであるGhidraでRustマルウェアを解析できるようにした．

課題

Rustは多くの関数が静的リンクされる．
→解析時に静的リンクされた関数の識別が困難
→人手による解析が必要な対象が増える

目的

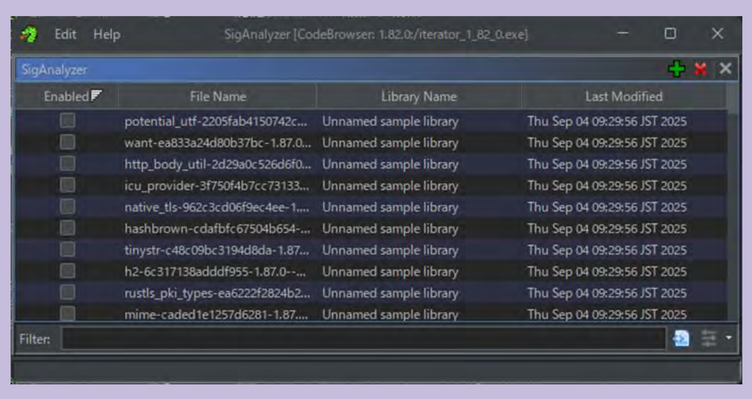
ライブラリ関数を自動識別
→**解析対象の関数を減らす**



プラグイン開発

SigAnalyzer

IDAのFLIRTシグネチャをGhidraにimportするプラグインを開発した．SigAnalyzerではFLIRTの一括適応，
登録したsignatureのon/offができる．



SigAnalyzerプラグイン

Microsoftが公開しているRust解析用プラグイン**RIFT**(Rust Interactive Function Tool)を**Ghidraで使えるようにした**．

RIFT実行前後の比較図(IDA)

既存ツールの評価検証

Rust製マルウェアを11検体収集し，調査を実施

評価1：シグネチャの生成に必要な情報であるCommit Hashがバイナリ内に残っているか調査

結果：Commit Hashが10/11残っていた
→ バイナリ内の文字列を活用してシグネチャの生成ができることを確認

評価2：RIFTをマルウェアに対して実行し，復元率を調査

結果：表の通り検知率が著しく低い
原因：LTO (Link Time Optimization)
→ 今後取り組む課題

検知率 = 名前の変わった関数 / 全関数

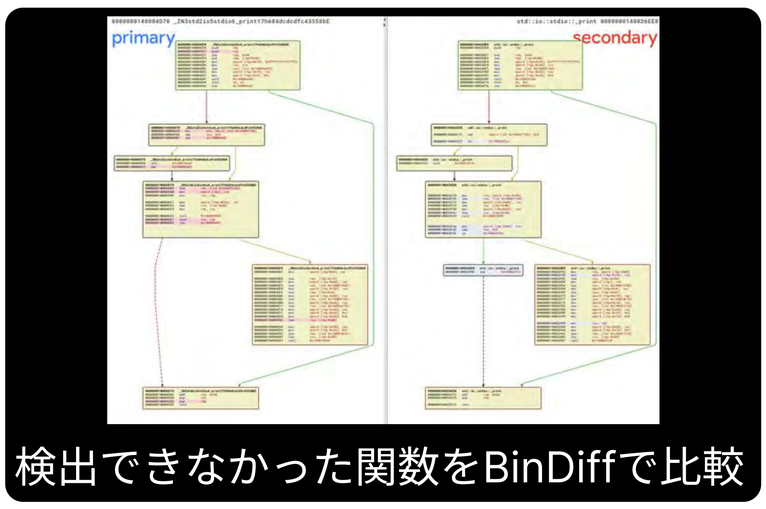
FLIRTでの検知率

Family Name	Detection Rate
Agenda(1)	1.7%
Agenda(2)	1.7%
Agenda(3)	4.6%
ALPHV	1.6%
FunkLocker	72%
HiveLocker	3.0%
SPICA	9.7%
RALord	78%
SSLoad(1)	38%
SSLoad(2)	6.4%
Akira_v2	20%

※CSS 2025で発表済み

今後

検知できなかった関数を**BinDiffで比較**したところ，**CFGが非常に似ていた**．既存の類似検知機能BSimはcos類似度を用いる手法である．精度を上げるために
CFGを用いた**グラフ理論ベースでの類似検知**機能を追加予定．



検出できなかった関数をBinDiffで比較