トップエスイー修了制作

抽象構文解析木(AST)を用いたハートブリード脆弱性の検査方法の検討と評価

フェリカネットワークス株式会社　礒崎 亮多　Ryota.Isozaki@FeliCaNetworks.co.jp

開発における問題と課題

問題　静的解析等のツールでは、ハートブリード脆弱性が一般的に広くサポートされてない

課題

[1] プログラムコードに脆弱性が含まれてないかどうか実装の妥当性を目視確認する必要がある

[2] 独自で自動検査ツールを実装したくとも、当該脆弱性の検査手法の報告がされてない

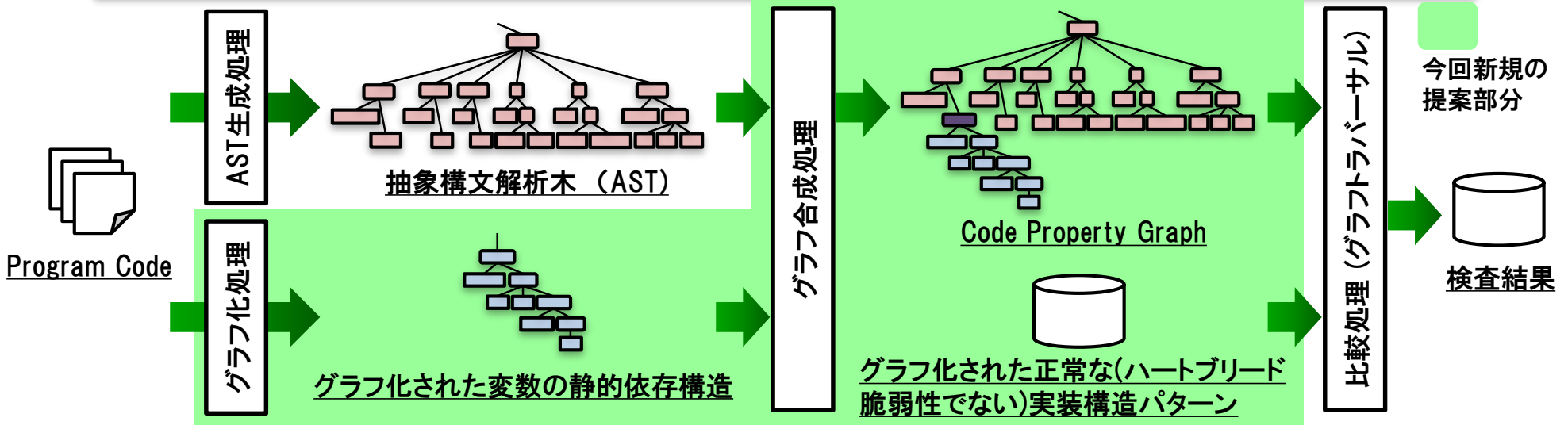
検査手法の提案による解決

ハートブリード脆弱性を自動検査するための手法(アルゴリズム)を提案し、課題を解決する

[1] 従来未検知であった問題を検知可能にする

[2] 誤検知を極力少なくする
(誤検知が多いと目視確認工数は低減されないため)

提案手法の概要

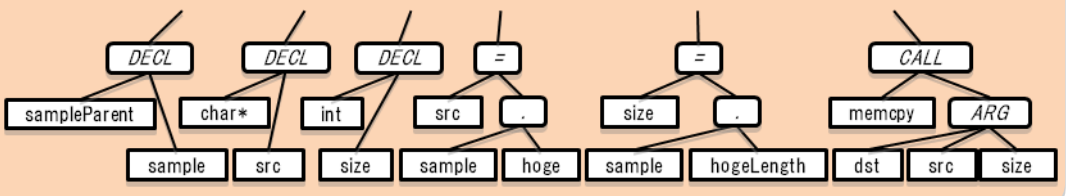


グラフ化された正常な(ハートブリード脆弱性でない)実装構造パターン (一例)

```
void sampleFunc()
{
  sampleParent  sample;
  char*         src;
  int           size;

                src = sample.hoge;
                size = sample.hogeLength;
                  …
                memcpy( dst, src, size );
                  …
}
```

```
typedef struct _sampleParent
{
    char*  hoge;
    int    hogeLength;
} sample;
```



openssl に対する評価結果および結論 (提案手法の効果)

① 従来「未検知」であったものについて、「検知」が可能となった

②「誤検知」は存在するが、目視確認の内容を約50%低減することが可能となった

分類	期待値	比較対象	提案手法
検知 (true positive)	2	0 (※1)	2
非検知 (true negative)	182	- (※2)	88
未検知 (false negative)	0	2 (※1)	0
誤検知 (false positive)	0	- (※2)	94
合計	184	-	1

	目視確認の箇所数	目視確認の時間	
提案手法を利用しない場合	184件 (全件)	2944min	
提案手法を利用する場合	2件 (検知箇所)	32min	1536min
	94件 (誤検知箇所)	1504min	
提案手法利用による効果		1408min(≒23.5hour)	

(※1) 従来手法や利用ツールでは検出不可のため　(※2) 同様の観点のアルゴリズムが存在しないため