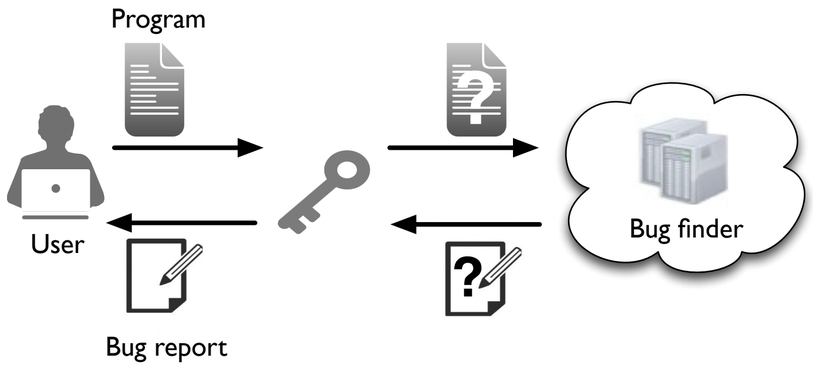
비밀 프로그램 분석

이우석[1], 홍현숙[2], 천정희[2], 이광근[1]
[1]서울대학교 컴퓨터공학부 프로그래밍 연구실
[2]서울대학교 수리과학부 암호학적 난제 연구단

목적

분석기와 대상 프로그램을 서로에게 숨기기



응용 가능한 예 :

- 상용 분석기 유료 온라인 서비스
- 앱 마켓 리뷰 시스템 (예 : 애플 앱스토어)

배경

동형암호(Homomorphic Encryption)

암호문에 연산한 것이 평문 연산 결과를 보존

$$\underline{op}(\mathcal{E}(m)) \equiv \mathcal{E}(op(m))$$

임의의 연산이 위 성질을 만족 시 완전동형암호

완전동형암호의 간단한 예

(공개키는 pq, 비밀키는 p, 평문은 이진수)

$$\mathcal{E}(m) = m + pq + 2\epsilon$$

p보다 작은 난수.
노이즈 누적의 원인

$$\mathcal{D}(c) = (c \bmod p) \bmod 2$$

이진 덧셈(XOR), 곱셈(AND) 보존하므로 완전동형

$$\mathcal{E}(m_1) + \mathcal{E}(m_2) \equiv \mathcal{E}(m_1 + m_2)$$
$$\mathcal{E}(m_1) \times \mathcal{E}(m_2) \equiv \mathcal{E}(m_1 \times m_2)$$

완전동형암호의 실용성

필요한 곱셈 횟수가 많을수록 실용성 급감.

비밀 포인터 분석

포인터 분석

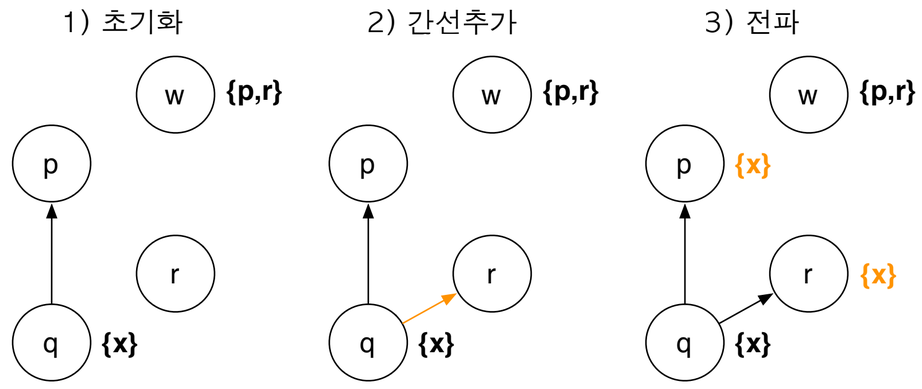
각 포인터 변수가 실행 중 가리킬 수 있는 주소들 분석
아래와 같이 도출되는 제약식을 만족시키는
$pt : Var \to 2^{Var}$ 계산.

명령어	제약식	
$x_i := \&x_j$	$\{x_j\} \subseteq pt(x_i)$	도입
$x_i := x_j$	$pt(x_j) \subseteq pt(x_i)$	복사
$x_i := *x_j$	$x_k \in pt(x_j)$ $pt(x_k) \subseteq pt(x_i)$	읽기(load)
$*x_i := x_j$	$x_k \in pt(x_i)$ $pt(x_j) \subseteq pt(x_k)$	쓰기(store)

그래프를 이용한 포인터 분석 알고리즘

각 노드는 포인터변수, 간선은 두 변수가 가리키는
주소집합 간 포함관계 표현

```
int x
int *p, *q, *r
int **w
q = &x
p = q
w = &p
w = &r
*w = q
```



초기화 후 2)간선추가 3)전파를 더 이상 변화가 없을 때까지 반복.

비밀 포인터 분석 - 분석에 필요한 모든 것 암호화 :

그래프, 각 변수가 가리키는 주소 집합(pt), 읽기/쓰기 명령어들

프로토콜 (m = #포인터변수, 이중 포인터타입($\tau **$)까지만 존재한다고 가정)

Definitions

$$\delta_{ij} \equiv \mathcal{E}(0) \iff x_i \notin pt(x_j)$$
$$\eta_{ij} \equiv \mathcal{E}(0) \iff pt(x_i) \not\subseteq pt(x_j)$$
$$u_{ij} \equiv \mathcal{E}(0) \iff \nexists x_i := *x_j$$
$$v_{ij} \equiv \mathcal{E}(0) \iff \nexists * x_i := x_j$$

Client inputs: For $1 \le i, j \le m$, ${}^1\delta_{ij}^{(1)}, {}^2\delta_{ij}^{(1)}, {}^1\eta_{ij}, {}^2\eta_{ij}, u_{ij}, v_{ij}$ where

$${}^2\delta_{ij}^{(1)} \equiv \begin{cases} \mathcal{E}(1) & \exists x_i := \&x_j \wedge x_j : \tau** \\ \mathcal{E}(0) & o.w \end{cases}$$
$${}^1\delta_{ij}^{(1)} \equiv \begin{cases} \mathcal{E}(1) & \exists x_i := \&x_j \wedge x_j : \tau* \\ \mathcal{E}(0) & o.w \end{cases}$$

${}^1\eta, {}^2\eta$ are similarly defined.

Step 1: propagation For $1 \le s, j, k \le m$

$${}^2\delta_{kj}^{(s+1)} \leftarrow {}^2\delta_{kj}^{s} + \textstyle\sum_{i \neq k,j} {}^2\delta_{ki}^{s} \cdot {}^2\eta_{ij}$$

Step 2: edge addition

(Load stmt) For $1 \le i, k \le m$
${}^1\eta_{ki} \leftarrow {}^1\eta_{ki} + \sum_{j \neq i,k} {}^2\delta_{kj} \cdot u_{ij}$

(Store stmt) For $1 \le j, k \le m$
${}^1\eta_{jk} \leftarrow {}^1\eta_{jk} + \sum_{i \neq j,k} {}^2\delta_{ki} \cdot v_{ij}$

Step 3: propagation For $1 \le s, j, k \le m$

$${}^1\delta_{kj}^{(s+1)} \leftarrow {}^1\delta_{kj}^{s} + \textstyle\sum_{i \neq k,j} {}^1\delta_{ki}^{s} \cdot {}^1\eta_{ij}$$

Output determination The client receives ${}^1\delta_{ij}^{(m+1)}, {}^2\delta_{ij}^{(m+1)} (1 \le i, j \le m)$ from the server and computes points-to sets as follows:

$$pt(x_j) = \begin{cases} \{x_i \mid \mathcal{D}({}^1\delta_{ij}^{(m+1)}) \neq 0\} & x_j : \tau** \\ \{x_i \mid \mathcal{D}({}^2\delta_{ij}^{(m+1)}) \neq 0\} & x_j : \tau* \end{cases}$$

실현 가능할 것으로 기대됨

- 한 암호문에 누적되는 곱하기 최대횟수 : (#포인터변수)^2
- 깊이 = $\log_2$ (최대 누적 곱 횟수)
- 일반적으로 **깊이 ≤ 50** 인 경우 실현가능한 수준으로 여겨짐

프로그램	LOC	#변수	**깊이**
bc-1.06	7,079	1,609	**10.7**
make-3.79.1	16,164	4,105	**12.0**
gawk-3.1	19,598	6,542	**12.7**