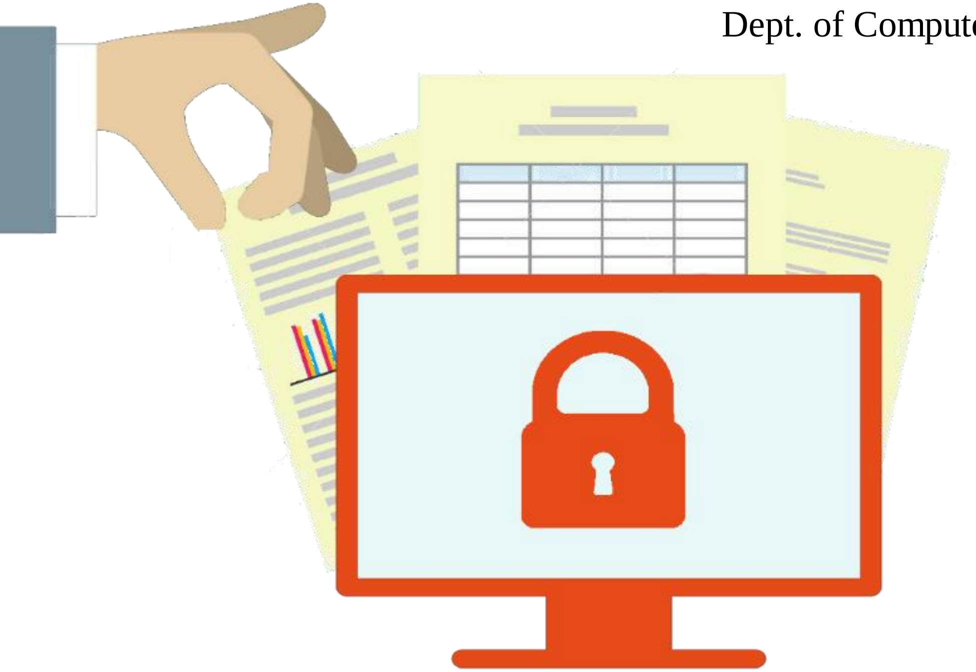
Quantifying Information Leakage of Probabilistic Programs Using the PRISM Model Checker

K. Salehi, A. A. Noroozi, and S. Amir-Mohammadian

Dept. of Computer Science, Shahrekord University, Shahrekord, Iran
Dept. of Computer Science, University of Tabriz, Tabriz, Iran
Dept. of Computer Science, University of the Pacific, Stockton, CA, USA



The Fifteenth International Conference on
Emerging Security Information, Systems and Technologies
SECURWARE 2021
Athens, Greece, November 14-18, 2021

IARIA

Authors

Khayyam Salehi
- Assistant professor
- Shahrekord University, Iran

Ali A. Noroozi
- Ph.D.
- University of Tabriz, Iran

Sepehr Amir-Mohammadian
- Assistant professor
- University of the Pacific, USA

Contents

1 **Introduction**

2 The proposed Method

3 Implementation and case study

4 Related work

5 Conclusion

Introduction

Confidentiality

Introduction

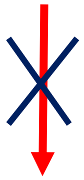
Common mechanisms for confidentiality:

Cryptography   Access control   Firewall

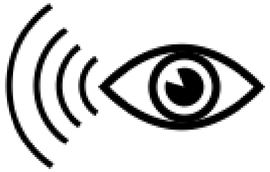
Introduction

Information leakage

secret variables



public variables



Information leakage

$$\texttt{l := h | } (1100)_b$$

2 rightmost bits of h are leaked into l

Introduction

Information leakage

```
while l1 < h mod 2 do
    l1 := l1 + 1;
    l2 := random(2);
od
```

1 bit of h is leaked into l1

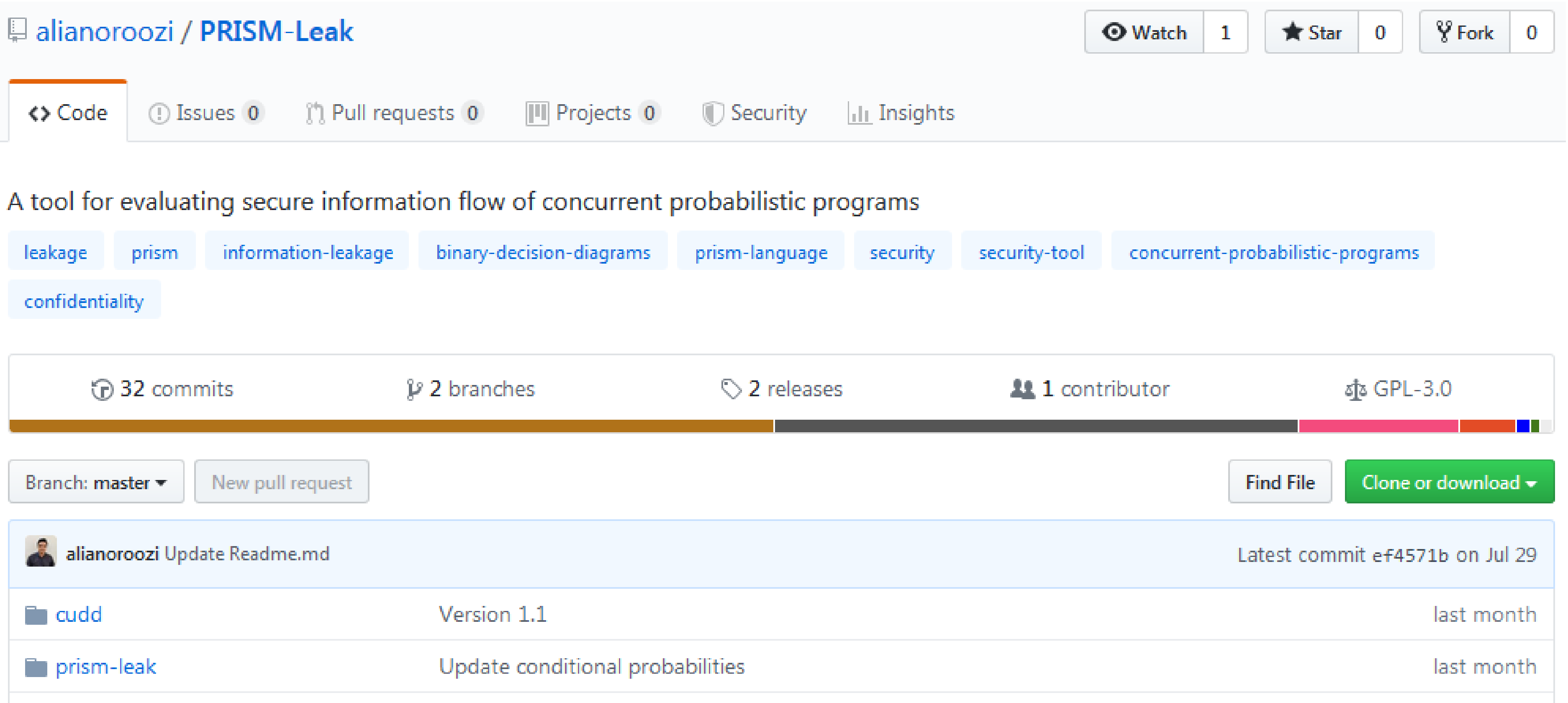
Contributions

1. An automated method:

- Modeling programs by Markov chains,
- Computing joint probabilities of the program's secrets and public outputs,
- Calculating the exact value of information leakage.

Contributions

2. PRISM-Leak



Contributions

3. Case study:

the grades protocol

Contents

1 Introduction

2 **The proposed Method**

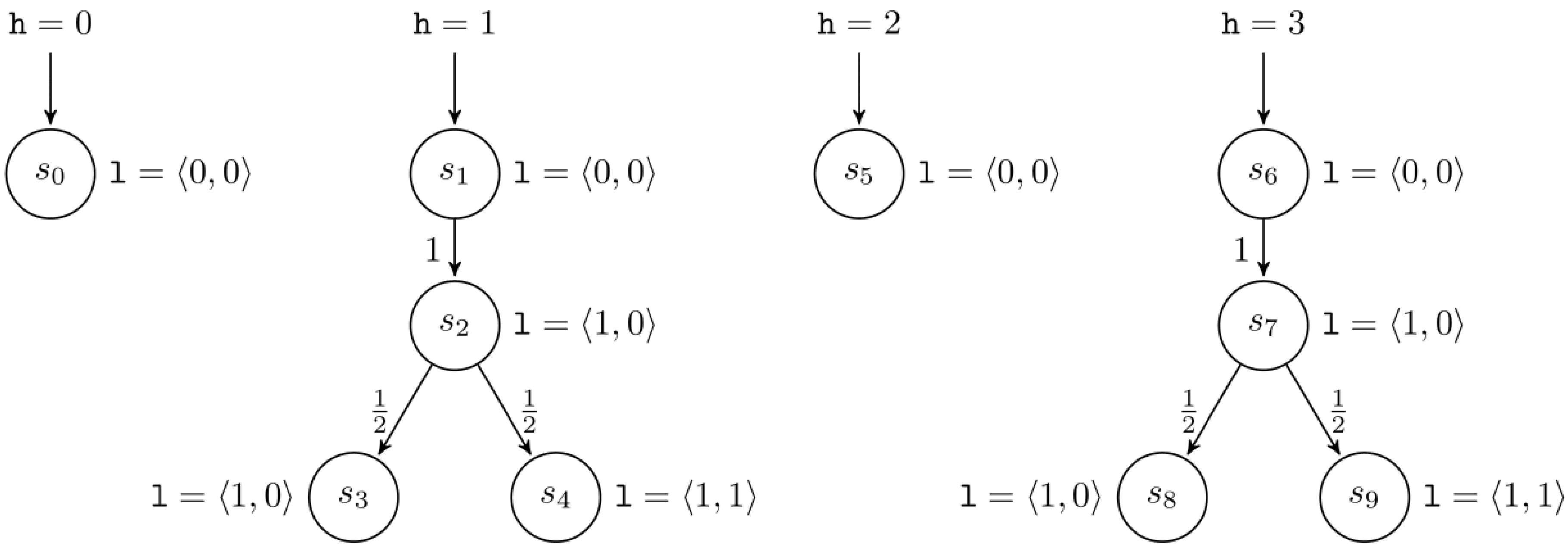
3 Implementation and case study

4 Related work

5 Conclusion

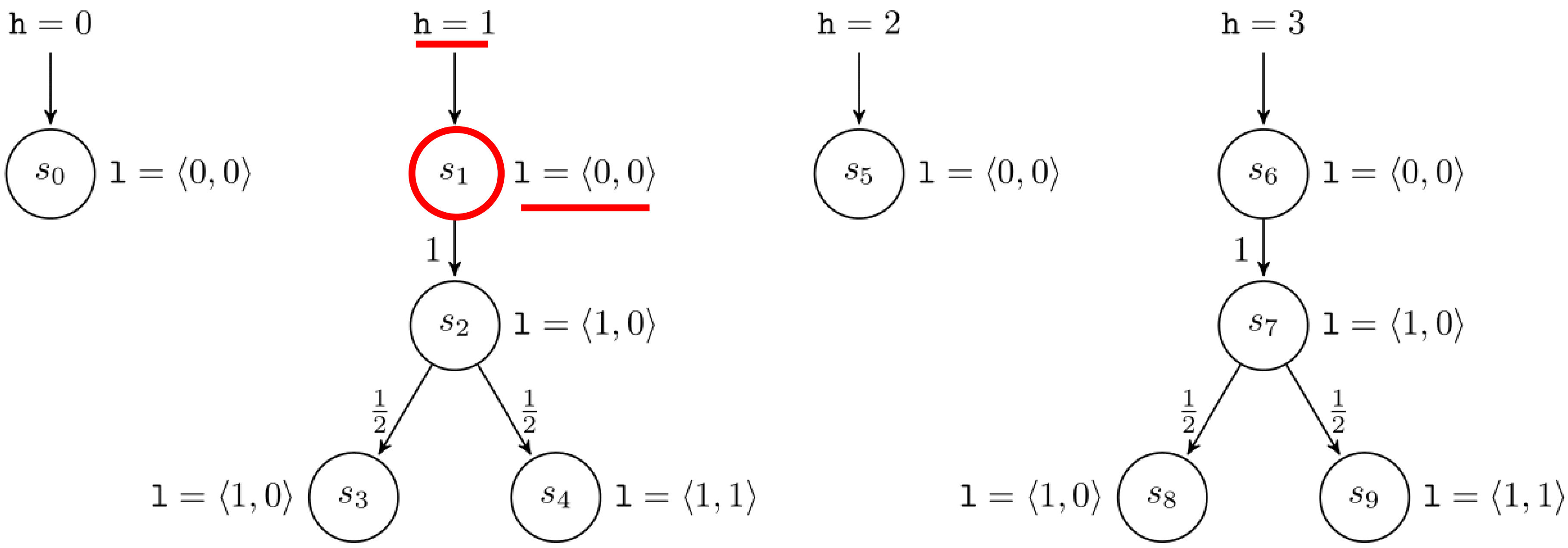
The proposed method

Markov Chain $\mathcal{M} = (S, \mathbf{P}, \zeta)$



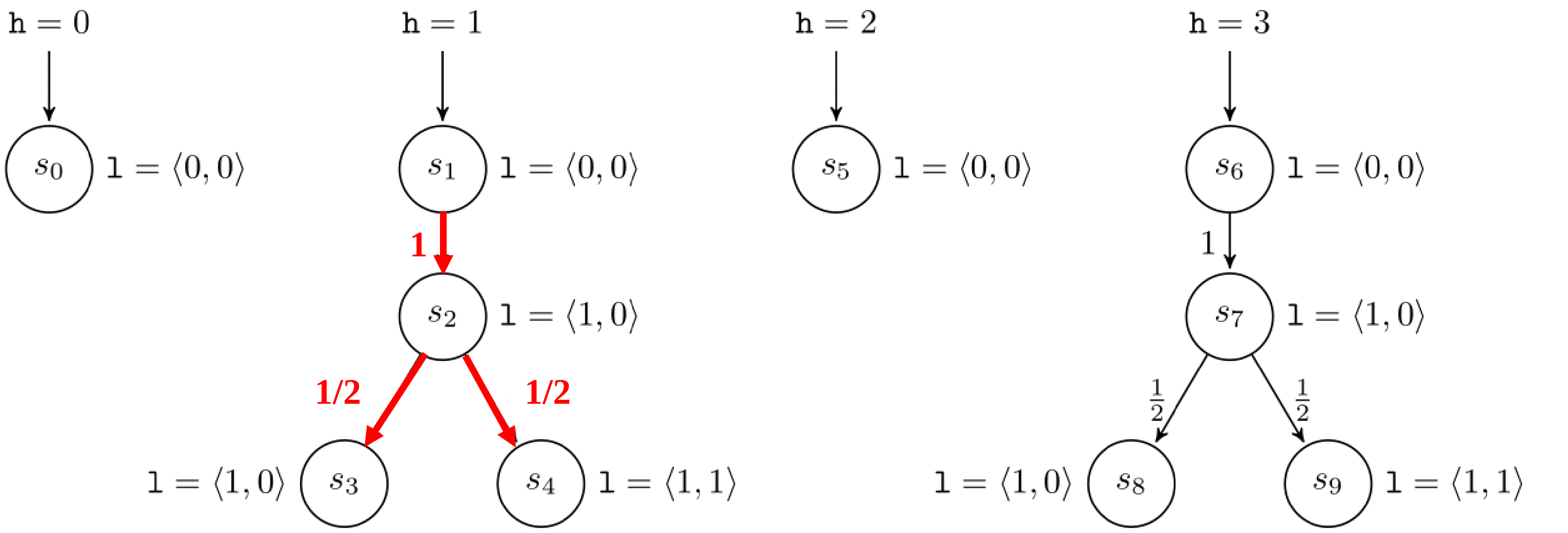
The proposed method

Markov Chain $\mathcal{M} = (S, \mathbf{P}, \zeta)$



3 The proposed method

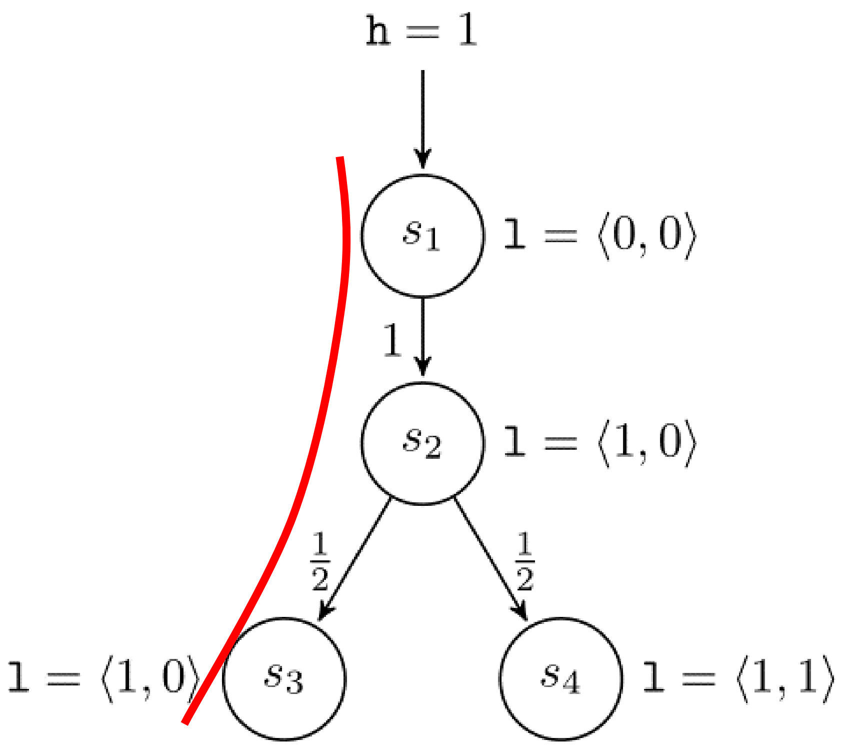
Markov Chain $\mathcal{M} = (S, \mathbf{P}, \zeta)$



Preliminaries

Path

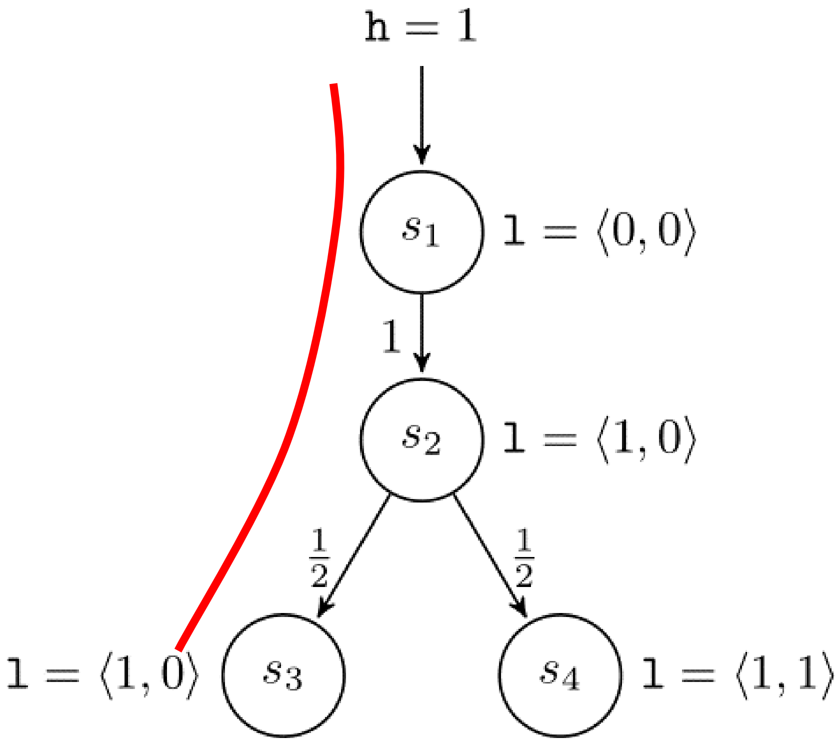
$\pi = s_1 s_2 s_3$



Preliminaries

Occurrence probability of a path

$$\Pr(\pi = s_1 s_2 s_3) = 0.25 * 1 * 0.5$$
$$= 0.125$$



The proposed method

Information leakage

$$\mathcal{L}(\mathcal{M}) = \text{initial uncertainty} - \text{remaining uncertainty}$$

3

The proposed method

Information leakage

$$\mathcal{L}(\mathcal{M}) = \text{initial uncertainty} - \text{remaining uncertainty}$$

Shannon entropy:

$$\mathcal{H}(\mathcal{X}) = -\Sigma_{x\in\mathcal{X}} Pr(\mathcal{X}=x)\ \log_2 Pr(\mathcal{X}=x)$$

The proposed method

Information leakage

$$\mathcal{L}(\mathcal{M}) = \text{initial uncertainty} - \text{remaining uncertainty}$$

$$\mathcal{L}(\mathcal{M}) = \mathcal{H}(h) - \mathcal{H}(h \mid o)$$

The proposed method

Initial uncertainty

$$\mathcal{H}(h) = -\sum_{\overline{h} \in h} Pr\left(h = \overline{h}\right).\log_2 Pr\left(h = \overline{h}\right)$$

The proposed method

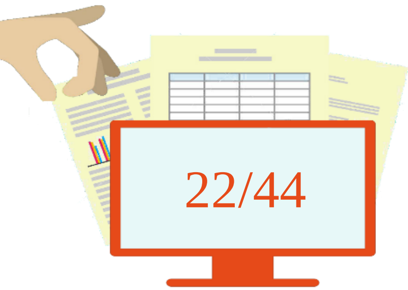
Remaining uncertainty

$$\mathcal{H}(h \mid o) = -\sum_{\overline{o} \in o} Pr(o = \overline{o}) \,.\, \mathcal{H}(h \mid o = \overline{o})$$

$$-\sum_{\overline{h} \in h} Pr\left(h = \overline{h} \mid o = \overline{o}\right).\log_2 Pr\left(h = \overline{h} \mid o = \overline{o}\right)$$

$$\frac{\Pr\left(h = \overline{h}, o = \overline{o}\right)}{\Pr(o = \overline{o})}$$

$$\sum_{\overline{h} \in h} Pr\left(h = \overline{h}, o = \overline{o}\right)$$



The proposed method

Remaining uncertainty

$$\mathcal{H}(h \mid o) = -\sum_{\bar{o} \in o} Pr(o = \bar{o}) \,.\, \mathcal{H}(h \mid o = \bar{o})$$

$$-\sum_{\bar{h} \in h} Pr\left(h = \bar{h} \mid o = \bar{o}\right) . \log_2 Pr\left(h = \bar{h} \mid o = \bar{o}\right)$$

$$\frac{\Pr\left(h = \bar{h}, o = \bar{o}\right)}{\Pr(o = \bar{o})}$$

$$\sum_{\bar{h} \in h} Pr\left(h = \bar{h}, o = \bar{o}\right)$$

The proposed method

$$\sum_{\bar{h} \in h} Pr\left(h = \bar{h}, o = \bar{o}\right) =$$

$$\sum_{s_0 \in Init(\mathcal{M}),\ s_n = <\bar{o},\bar{h},.,.>} Pr(\pi = s_0 \dots s_n)$$

The proposed method

Input: finite MC $\mathcal{M}$

Output: a map containing the joint probabilities $Pr(h, o)$

Let $ohMap$ be an empty higher-order map function from $\overline{o}$ to $\overline{h}$ to $Pr(h = \overline{h}, o = \overline{o})$;
// i.e. $ohMap : \overline{o} \mapsto (\overline{h} \mapsto Pr(h = \overline{h}, o = \overline{o}))$
Let π be an empty list of states for storing a path;
for s_0 **in** $Init(\mathcal{M})$ **do**
    EXPLOREPATHS(s_0, π, $ohMap$);
return $ohMap$;

The proposed method

```
function EXPLOREPATHS(s, π, ohMap)
    // add state s to the current path from the initial state
    π.add(s);
    // found a path stored in π
    if s is a terminating state then
        // assume s = ⟨ō, h̄, ·, ·⟩
        // define hMap as Pr(h, o = ō)
        if ō not in ohMap then
            Let hMap be an empty map from
                              h̄ to Pr(h = h̄, o = ō);
        else
            hMap = ohMap.get(ō);
        if h̄ not in hMap then
            prob = Pr(π);
        else
            prob = Pr(π) + hMap.get(h̄);
        hMap.put(h̄, prob);  // Update hMap
        ohMap.put(ō, hMap);  // Update ohMap
    else
        for s′ in Post(s) do
            EXPLOREPATHS(s′, π, ohMap);
    // done exploring from s, so remove it from π
    π.pop();
    return ;
```

3 The proposed method

Time complexity:

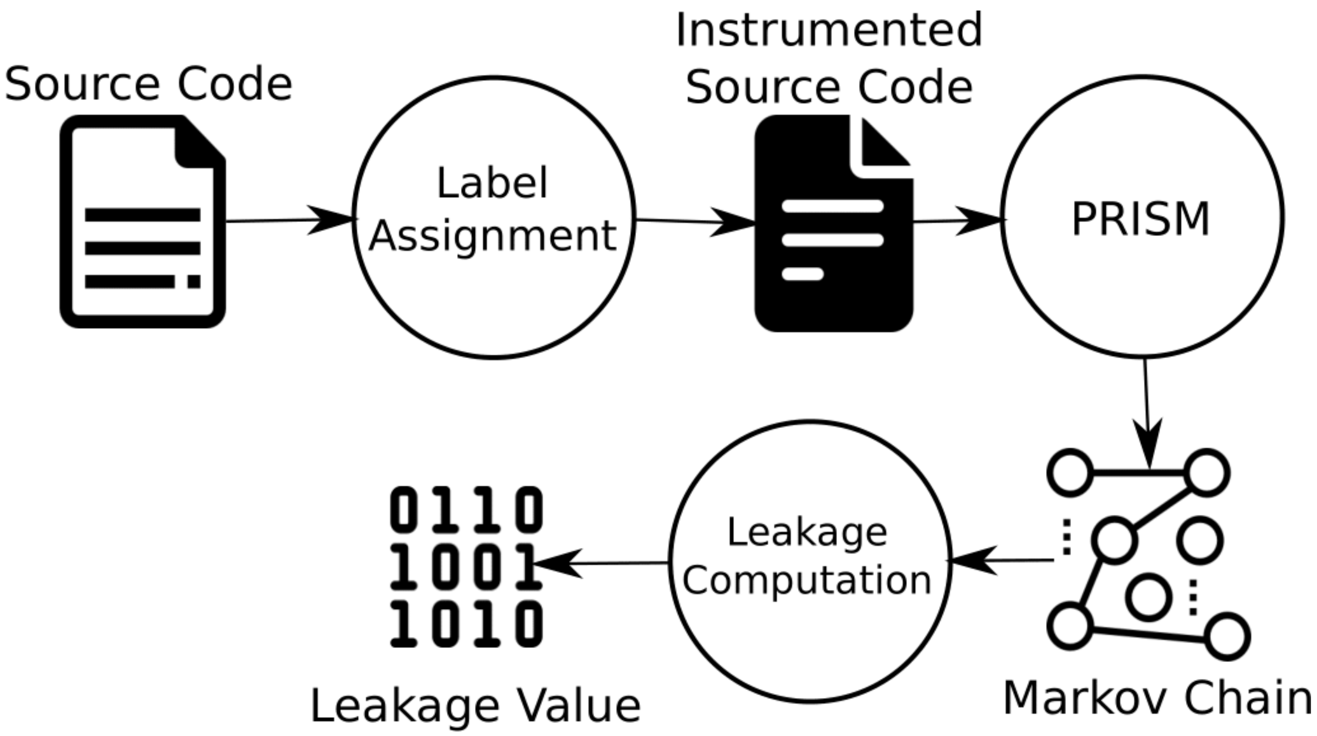
$$O(2^n)$$

Contents

1. Introduction
2. The proposed Method
3. **Implementation and case study**
4. Related work
5. Conclusion

Implementation

PRISM-Leak:



Case study

The grades protocol

- k students $s_1, \dots, s_k$
- secret grades $g_1, \dots, g_k$ where $0 \leq g_i < m$

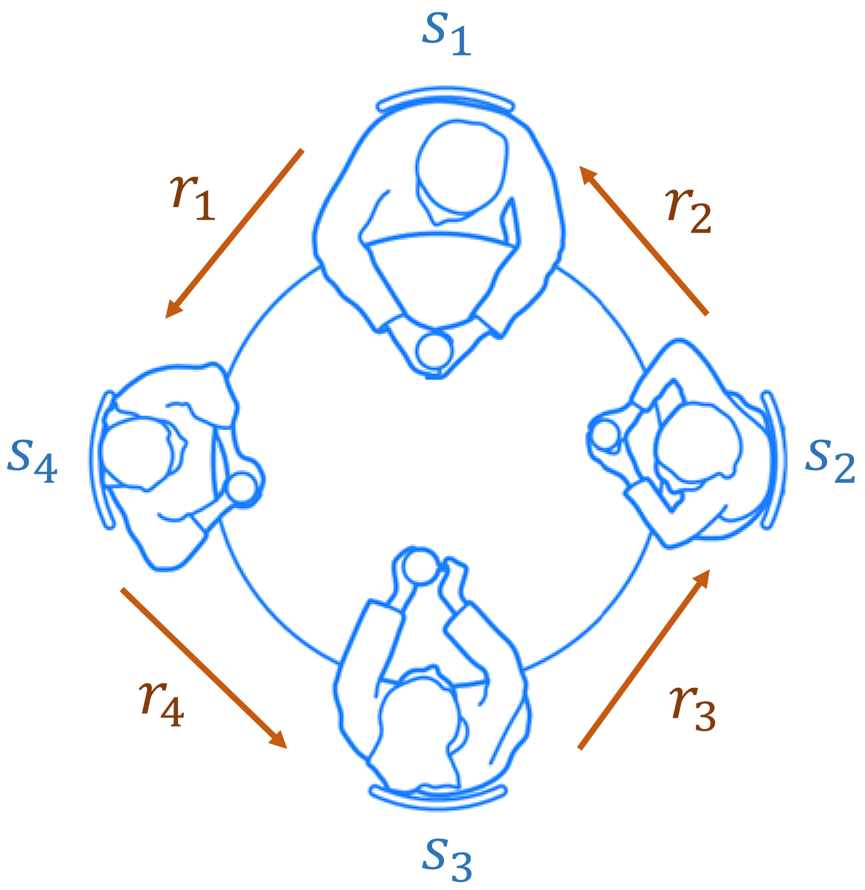
- Goal: computing sum of the grades, without revealing the secret grades to other students

Case study

The grades protocol

- k students $s_1, \ldots, s_k$
- secret grades $g_1, \ldots, g_k$ where $0 \leq g_i < m$

- $n = (m-1) \times k + 1$
- $r_i \in [0, n]$
- $d_i = g_i + r_i - r_{(i+1)\%k}$
- sum $= \left(\sum_i d_i\right) \% \, n$



Case study

The grades protocol

m	k	The grades protocol: $\mathcal{M}_{grades}$ # states	The grades protocol: $\mathcal{M}_{grades}$ # transitions	The grades protocol: Leakage (bits)	The sum of the grades: $\mathcal{M}_{sum}$ # states	The sum of the grades: $\mathcal{M}_{sum}$ # transitions	The sum of the grades: Leakage (bits)
2	2	196	228	1.5 (75%)	16	20	1.5
	3	3752	4256	1.81 (60.4%)	64	104	1.81
	4	92496	102480	2.03 (50.8%)	256	528	2.03
3	2	1179	1395	2.2 (69.3%)	36	45	2.2
	3	66366	75600	2.53 (53.1%)	216	351	2.53
	4	439668	597780	2.75 (43.3%)	1296	2673	2.75
4	2	4048	4816	2.66 (66.4%)	64	80	2.66
	3	455104	519040	2.98 (49.7%)	512	832	2.98
	4	3271680	6589440	3.2 (40%)	4096	8448	3.2

Contents

1 Introduction

2 The proposed Method

3 Implementation and case study

4 **Related work**

5 Conclusion

Related work

Chothia et al., 2013

- Tool LeakWatch
- Java programs
- Estimation of the leakage
- Intermediate leakages

Related work

Klebanov, 2014

- Symbolic execution and self-composition
- Deterministic programs
- Non-automated method

Related work

Biondi et al., 2017

- Tool HyLeak
- Sequential programs
- Estimation of the leakage
- No intermediate leakage

4

Related work

Salehi et al., 2019

- Evolutionary algorithm
- Channel capacity
- Concurrent probabilistic programs

Contents

1. Introduction
2. Preliminaries
3. The proposed method
4. Related work
5. **Conclusion and future work**

Conclusion

Proposed approach:

Formal ✓

Fully-automatic ✓

Accurate ✓

Future work

1. Comparing scalability

2. Estimating leakage by statistical methods

3. Analyzing case studies in other application domains

References

[1] "CWE-200: Exposure of Sensitive Information to an Unauthorized Actor," https://rb.gy/ac6ui0, [retrieved: 10, 2021].

[2] "OWASP Top 10 Privacy Risks," https://rb.gy/vhq4qj, [retrieved: 10, 2021].

[3] A. Sabelfeld and A. C. Myers, "Language-based information-flow security," *IEEE J-SAC*, vol. 21, no. 1, pp. 5–19, 2003.

[4] G. Smith, "Principles of secure information flow analysis," in *Malware Detection. Advances in Information Security, vol 27*. Springer-Verlag, 2007, pp. 291–307.

[5] M. R. Clarkson and F. B. Schneider, "Hyperproperties," *J. Comput. Secur.*, vol. 18, no. 6, pp. 1157–1210, 2010.

[6] D. Schoepe, M. Balliu, B. C. Pierce, and A. Sabelfeld, "Explicit secrecy: A policy for taint tracking," in *EuroS&P*. IEEE, 2016, pp. 15–30.

[7] C. Skalka, S. Amir-Mohammadian, and S. Clark, "Maybe tainted data: Theory and a case study," *J. Comput. Secur.*, vol. 28, no. 3, pp. 295–335, April 2020.

[8] M. S. Alvim, M. E. Andrés, K. Chatzikokolakis, P. Degano, and C. Palamidessi, "Differential privacy: On the trade-off between utility and information leakage," in *FAST*. Springer, 2011, pp. 39–54.

[9] P. Cuff and L. Yu, "Differential privacy as a mutual information constraint," in *CCS*, 2016, pp. 43–54.

[10] F. Biondi and et al., "Scalable approximation of quantitative information flow in programs." in *VMCAI*, 2018, pp. 71–93.

References

[11] M. Jurado, C. Palamidessi, and G. Smith, "A formal information-theoretic leakage analysis of order-revealing encryption," in *CSF*. IEEE Computer Society, 2021, pp. 1–16.

[12] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press Cambridge, 2008.

[13] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2006.

[14] M. S. Alvim and et al., *The Science of Quantitative Information Flow*. Springer, 2020.

[15] V. Klebanov, "Precise quantitative information flow analysis—a symbolic approach," *Theor. Comput. Sci.*, vol. 538, pp. 124–139, 2014.

[16] F. Biondi, Y. Kawamoto, A. Legay, and L.-M. Traonouez, "Hyleak: hybrid analysis tool for information leakage," in *ATVA*. Springer, 2017, pp. 156–163.

[17] A. A. Noroozi, J. Karimpour, and A. Isazadeh, "Information leakage of multi-threaded programs," *Comput. Electr. Eng.*, vol. 78, pp. 400–419, 2019.

[18] R. Chadha, U. Mathur, and S. Schwoon, "Computing information flow using symbolic model-checking," in *FSTTCS*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014, pp. 505–516.

[19] A. Weigl, "Efficient sat-based pre-image enumeration for quantitative information flow in programs," in *DPM*. Springer, 2016, pp. 51–58.

[20] M. S. Alvim and et al., "An axiomatization of information flow measures," *Theor. Comput. Sci.*, vol. 777, pp. 32–54, 2019.

[21] R. Pardo, W. Rafnsson, C. Probst, and A. Wasowski, "Privug: Quantifying leakage using probabilistic programming for privacy risk analysis," *arXiv preprint arXiv:2011.08742*, 2020.

References

[22] F. Biondi, A. Legay, P. Malacaria, and A. Wasowski, “Quantifying information leakage of randomized protocols,” *Theor. Comput. Sci.*, vol. 597, no. C, pp. 62–87, 2015.

[23] S. Amir-Mohammadian, “A semantic framework for direct information flows in hybrid-dynamic systems,” in *CPSS-AsiaCCS*. ACM, June 2021, pp. 5–15.

[24] A. A. Noroozi, K. Salehi, J. Karimpour, and A. Isazadeh, “Prism-leak - a tool for computing information leakage of probabilistic programs,” https://rb.gy/elgkyi, [retrieved: 10, 2021].

[25] M. Kwiatkowska, G. Norman, and D. Parker, “PRISM 4.0: Verification of probabilistic real-time systems,” in *CAV*. Springer, 2011, pp. 585–591.

[26] C.-D. Hong, A. W. Lin, R. Majumdar, and P. Rümmer, “Probabilistic bisimulation for parameterized systems,” in *CAV*. Springer, 2019, pp. 455–474.

[27] D. Parker, “Implementation of symbolic model checking for probabilistic systems,” Ph.D. dissertation, University of Birmingham, 2002.

[28] M. Backes, B. Köpf, and A. Rybalchenko, “Automatic discovery and quantification of information leaks,” in *S&P*. IEEE, 2009, pp. 141–153.

[29] T. Chothia, Y. Kawamoto, C. Novakovic, and D. Parker, “Probabilistic point-to-point information leakage,” in *CSF*. IEEE, 2013, pp. 193–205.

[30] T. Chothia, Y. Kawamoto, and C. Novakovic, “Leakwatch: Estimating information leakage from java programs,” in *ESORICS*. Springer, 2014, pp. 219–236.

[31] K. Salehi, J. Karimpour, H. Izadkhah, and A. Isazadeh, “Channel capacity of concurrent probabilistic programs,” *Entropy*, vol. 21, no. 9, p. 885, 2019.

[32] A. A. Noroozi, K. Salehi, J. Karimpour, and A. Isazadeh, “Secure information flow analysis using the prism model checker,” in *ICISS*. Springer, 2019, pp. 154–172.

Thanks for you attention!