数学的技法にみる情報セキュリティの安全性解析手法

研究ノート

矢　内　直　人*

Security Analysis Methods for Information Security by Mathematical Proofs

Key Words：Information Security, Security Analysis, Formal Methods, Provable Security

1．はじめに

情報セキュリティ技術の最大の特徴は「安全性」である。今使われている技術はどのような安全性を持つのか、どのように高い安全性をなし得るのか、情報セキュリティにおける研究は内容を問わず何らかの観点で安全な技術を探求することにある。しかしながら、この「安全性」という概念は何を意味するのかが最も複雑かつ難しい議論になる。これは「安全性」が主観的かつ抽象的になりがちな概念のためである。

この主観的かつ抽象的な問題をかみ砕いた例として、セキュリティソフトを購入する状況を考えよう。ソフト購入時に「A社製の製品は良い」という尺度を用いる人も少なからず存在する。このとき、そういった人はA社への個人的な信頼という主観で購入に踏み切っており、（少なくとも上述した字面だけの判断基準においては）本人の経験則に基づく判断といえる。このような判断基準は客観的かつ具体的な指標でそのソフトの良さを保証し得るとは決して限らない。とくに、周囲の人から見たとき、何故A社に信頼を置いているかはときとして非常に曖昧である。

筆者は上述した「安全性」という概念について、具体的かつ客観的な指標で解析・評価する方法について研究してきた。本稿ではその手法と当該分野における近年の動向について紹介する。なお、本研究はJST ACT-I（http://www.jst.go.jp/kisoken/act-i/）採択課題「暗号学的期待値分布に基づくソースコードレベルでの汎用的脆弱性検証手法」に採択された内容に基づいている。

2．安全性の解析方法

安全性を客観的に議論し、かつ、具体的に解析するには、どのような手法が良いだろうか。最も多くの人が思い浮かぶ方法は、攻撃ツールを用いることで攻撃が成功するか確認するペネトレーションテストと呼ばれる方法である。この方法は攻撃が実際に成功するか否かという判断できる点で、具体的な解析と個々の結果に関する客観的な指標を持つことが可能である。一方で、ありとあらゆる攻撃パターンを試す必要があるため、防ぎたい攻撃の数だけテストを行わなければならない。また、確認済みの攻撃に極めて類似した攻撃であっても、未テストの攻撃に対しては安全性が不明という問題もある。

現在最も関心を集めている安全性の解析手法は、証明など数学的検証を通じた方法である。これは安全性を述語論理やチューリングマシンなど数学的に表現することで、安全性解析を数学の概念に落とし込むことが可能となる。このとき、数学的正しさの確認を通じて安全性解析の正しさを確認することが可能となる。従来の情報セキュリティの研究では、この数学的検証は各研究者が自らの手で数式や証明をノートに書き下す形で行われていた。近年の研究ではこの検証を計算機に行わせることで安全性解析を自動化させる研究が主流となりつつある。以降ではそれらの詳細を説明する。

* Naoto YANAI

1987年3月生まれ
筑波大学大学院　システム情報工学研究科　リスク工学専攻博士後期課程（2014年）
現在、大阪大学　大学院情報科学研究科　マルチメディア工学専攻　セキュリティ工学講座　助教　博士（工学）　情報セキュリティ
TEL：06-6879-4517
FAX：06-6879-4519
E-mail：yanai@ist.osaka-u.ac.jp

3．ソフトウェア工学における形式手法との出会い

安全性解析を自動化させるきっかけとなったものは、ソフトウェア工学分野における「形式検証」との出会いである。従来の形式検証は設計システムが仕様を満たしているか数学的に検証するものであった。この仕様の検証として安全性を対象にすることが、形式検証を用いる安全性解析の直観である。

形式検証には大きく分けて定理証明器とモデル検査の二つの方法がある。定理証明器は形式的な論理体系の中で定理を導く作業を補助し、その正しさを確認することができる。具体的なソフトウェアとしてはCoq [1)] やIsabelle [2)] が挙げられる。一方、モデル検査では形式的な論理体系の中で取りうる状態を網羅的に探索する。具体的なソフトウェアとしてはAlloy [3)] やProVerif [4)] などが挙げられる。情報セキュリティの安全性解析で考えた場合、一般には前者の定理証明器は形式的に証明をする際に有効であり、後者はプロトコルに対する具体的攻撃を発見する際に有効といえる。

当該分野における情報セキュリティと形式検証を融合した近年の代表的研究成果としては、Coqで実装の正しさを証明するJasmin [1] や、Alloyで設計したWebシステムの安全性を評価できる汎用フレームワーク [2] などが知られている。また、フランスの国立研究機関INRIA [5)] とマイクロソフトを中心に進めているエベレスト・プロジェクト [6)] ではHTTPSなど様々な暗号化通信技術の安全性を形式検証で解析した実装・標準化を進めている。加えて、IETF [7)] などインターネット技術の標準化を議論する団体においても形式検証の議論がされる [3] など、産業分野でも高い注目がされている。

4．従来研究に見る暗号の証明との対比

前節で述べたとおり、従来の情報セキュリティでは安全性解析を手書きの証明で行っていた。とくに暗号技術では証明可能安全性と呼ばれる、暗号の安全性を数学的に証明する概念が知られていた。これは攻撃者が暗号を破る際に現在の計算機能力では解けないはずの問題を解かなければならないことを証明することで、安全性を保証するものである。これは前節で述べた形式検証に極めて近い背景を持つ概念といえる。

1) https://coq.inria.fr/
2) https://isabelle.in.tum.de/
3) http://alloytools.org/
4) http://prosecco.gforge.inria.fr/personal/bblanche/proverif/
5) Institut National de Recherche en Informatique et en Automatique, https://www.inria.fr/
6) https://project-everest.github.io/
7) Internet Engineering Task Force, https://www.ietf.org/

形式検証と証明可能安全性に関する最大の違いは、安全性の定義に紐づく数学的な概念そのものにある。形式検証、とくにモデル検査では安全性は記号モデルと呼ばれる、攻撃者のインターフェースとできる計算のみを列挙し、また、攻撃者が攻撃に成功する可能性（確率）を考慮しない世界で議論する。一方、証明可能安全性では計算論的モデルと呼ばれる世界で議論する。この世界では攻撃者は確率的多項式時間チューリング機械で定義され、どのような計算を内部で行っているかはとくに考慮しない。また、その際の実行時間と攻撃の成功確率はセキュリティパラメータと呼ばれるパラメータに依存して決定される。すなわち、達成したい安全性に対して具体的に定義される。（少々乱暴な分類ではあるが）大ざっぱに要点を整理すると、形式検証では「攻撃者が行う処理のみを列挙し、目標要件は考慮しない」一方、証明可能安全性では「攻撃者が何をするかは気にせず、目標要件は具体化」しているといえる。

上述した観点からは証明可能安全性（すなわち計算論的モデル）の議論が優れているように見える。しかしながら、形式検証は前述したとおり機械に検証させることで解析を自動化できる点に大きな利点がある。また、計算論的モデルでの証明は非常に煩雑であるため、論文が公開された後に誤りが報告された事例も少なくない。

以上を踏まえると本当に望ましいものは両方の利点を融合させること、すなわち、計論的モデルで解析を自動化させることである。これは計算論的モデルを通じた高い安全性を誤りなく効率的に示せることを意味している。次節以降で、この両方のモデルを統合した代表的成果を紹介する。

5．統合的フレームワーク

記号モデルと計算論的モデルの二つを統合した初の成果はAbadiとRogawayによって示された[4]。彼らは暗号技術の安全性が記号モデルで証明された場合、計算論的モデルでの安全性も証明できることを保証する記号モデルでの計算論的健全性を示している。前節で述べたとおり、記号モデルでの安全性は様々な形式手法を通じて自動化ができるため、この成果は計算論的モデルでの安全性解析を間接的に自動化できることを意味している。

AbadiとRogawayが解析した安全性は暗号の受動的攻撃者と呼ばれる暗号化された通信内容を傍受するだけの攻撃者に対する弱い安全性であった。後発の研究により同様の試みは発展がなされ、近年ではDahlとDamgård[5]が汎用的結合可能性と呼ばれる、どのような使われ方をしても安全性が保証できる強い性質の証明を、記号モデルを通じて自動化できることを示している。DahlとDamgårdの成果はとくに二者間で計算される多くの暗号技術に適用できる汎用的な成果であり、当該分野に与えた影響は大きいと筆者は考えている。

また、3節で述べたとおりエベレスト・プロジェクトではINRIAやマイクロソフトを中心に、安全性を証明した暗号化通信プロトコルをInternet Exploreに導入するなど、産学の溝を埋める成果を多数挙げている。とくに文献[6]ではC言語により記述したハッシュ関数などのプログラムを形式検証の言語に変換することで、実装状態から安全性を証明可能である。これは将来的に任意のプログラミング作業に対して安全性を証明できるなど、高い可能性を感じさせる成果といえる。

6．安全性解析のこれから

現在、情報セキュリティの安全性解析の自動化は著しい発展を見せている。従来は暗号技術の安全性の証明が中心であったが、エベレスト・プロジェクトに見られるように、プログラムコードに対して証明を与えることで安全性を解析する成果も出始めている。近い将来に、例えばユーザがスマートフォンにインストールしたアプリケーションを、ソースコードに対して証明を得られるか検討することで安全性を解析する時代の到来も起こり得る。また、本稿では紙面上割愛したが、形式検証の発展を通じて新たなサイバー攻撃の発見にも使われつつある。今後は形式検証を用いて安全性解析はどこまでできるか、また、どういう機能・脅威に備えて議論していくかが大きな要点となる。

参考文献

[1] José Bacelar Almeida, Manuel Barbosa, Gilles Barthe, Benjamin Grégoire, Vincent Laporte, Tiago Oliveira, Hugo Pacheco, Benedikt Schmidt, Pierre-Yves Strub “Jasmin: High-Assurance and High-Speed Cryptography”, Proc. of CCS 2017, pp. 1807-1823, ACM, October-November 2017.

[2] Devdatta Akhawe, Adam Barth, Peifung E. Lam, John Mitchell, Dawn Song, “Towards a Formal Foundation of Web Security”, Proc. of CSF 2010, pp. 290-304, IEEE, July 2010.

[3] Cas Cramer, and Jonathan Hoyland, “Exported Authenticators”, IETF, March 2018. http://www.brookings.edu/~/media/Files/rc/papers/2011/06_barcelona_metro_innovation/06_barcelona_22_presenation.pdf

[4] Martín Abadi, Phillip Rogaway, “Reconciling Two Views of Cryptography (The Computational Soundness of Formal Encryption)”, Journal of Cryptology, 15(2), pp.103-127, Sprigner, January 2002.

[5] Morten Dahl, Ivan Damgård, “Universally Composable Symbolic Analysis for Two-Party Protocols Based on Homomorphic Encryption”, Proc. of EUROCRYPT 2014, LNCS 8441, pp.695-712, Springer, May 2014.

[6] Jean-Karim Zinzindohoué, Karthikeyan Bhargavan, Jonathan Protzenko, Benjamin Beurdouche, “HACL*: A Verified Modern Cryptographic Library”, Proc. of CCS 2017, pp. 1789-1806, ACM, October-November 2017.