1976

Proving Protection Systems Safe

D. E. Denning

P. J. Denning

S. J. Garland

M. A. Harrison

W.L. Ruzzo

Report Number:
76-209

PROVING PROTECTION SYSTEMS SAFE

D. E. Denning[1]

P. J. Denning[1]

S. J. Garland[2]

M. A. Harrison[3]

W. L. Ruzzo[3,4]

November 1976

Revised February 1978

CSD TR 209

Abstract

Theories of protection powerful enough to resolve security questions of computer systems are considered. No single theory of protection is adequate for proving whether the security afforded by an arbitrary protection system suffices to safeguard data from unauthorized access. When theories of protection are restricted to computer systems which are bounded in size, adequate theories exist, but they are inherently intractable. The implications of these results are discussed.

[1]Computer Science Department, Purdue University, W. Lafayette, IN 47907. Research sponsored by the National Science Foundation MCS75-21100.

[2]Department of Mathematics, Dartmouth College, Hanover, NH 03755. Research sponsored by the Office of Naval Research Contract ONR N00014-75-C-0514.

[3]Computer Science Department, University of California, Berkeley, CA 94720. Research sponsored by the National Science Foundation MCS74-07636-A01.

[4]Current address: Department of Computer Science, University of Washington, Seattle, Washington 98195. Research supported by an IBM Pre-doctoral Fellowship.

1. Introduction

No existing computer system is completely secure. None has withstood penetration. None can truthfully claim complete protection of files and other confidential user objects. Most computers are, in fact, easily compromised. Have designers somehow paid insufficient attention to security policies and mechanisms? Or is it inherently difficult to determine whether a given security policy and mechanism is safe, that is, whether it can provide a proper degree of safety for user objects?

We consider the prospects for developing a comprehensive theory of protection (or even a finite number of such theories) sufficiently general to enable proofs or disproofs of safety for arbitrary protection systems. Not surprisingly, we can show that there is no decidable theory adequate for proving all propositions about safety. To bring these results closer to practice, we also consider theories of protection for computer systems which are bounded in size. Here there are decidable theories adequate for proving all propositions about safety. However, proving safety intrinsically requires enumerating all conceivable ways in which safety may be compromised, and hence may use enormous amounts of time or space to reach a conclusion.

These results must be interpreted carefully. They are about the fundamental limits of our abilities to demonstrate that protection systems are safe. They do not rule out the

possibility that we can construct individual protection systems and prove that they are safe, or that we can find practical restrictions on protection systems which make safety questions tractable. Yet, these results do suggest that systems without severe restrictions on their operation will have safety questions too expensive to answer. Thus we may be forced to shift our concern from proving whether or not protection systems are completely safe to proving whether breaches in security are sufficiently costly to deter concerted attacks.

Our results are obtained by applying known results from logic and complexity theory to questions of safety for protection systems. The transferability of such results shows that questions concerning safety are much more difficult to answer than many might have believed.

2. Access control models of protection systems

Protection systems encompass two kinds of policies and mechanisms for enforcing them. Access control policies regulate the right of access to objects in the system; flow control policies regulate the dissemination of information among the objects of the system. Since flow cannot occur without access, flow control problems are at least as difficult as access control problems. Accordingly, we will limit ourselves in this paper to safety problems with access control policies.

An access control policy specifies a set of objects, a set of domains of access to these objects, a set of generic rights to objects, and a method of binding processes to domains. An access control state is defined by a given distribution of generic rights; it can be envisaged as an access matrix whose rows correspond to domains and columns to objects. A process operating in domain d may use an object x in a particular way only if a generic right enabling that action appears in the access matrix at position (d,x). Some rights govern when processes may change the access control state.

A protection system comprises an access control policy together with an enforcement mechanism for the policy. Enforcement mechanisms are judged by the criteria of security and precision. A mechanism is secure if it disables all operations prohibited by the policy; it is precise if it is secure and enables all operations permitted by the policy. The number of permitted operations disabled is a measure of a mechanism's imprecision. Whereas most systems may be able to tolerate some lack of precision, few can tolerate a lack of security.

The users of a system demand more than security or precision. They demand that both policy and mechanism be safe; that is, no process should be able to acquire a right to an object in violation of the declared intentions of the object's owner. An example of an unsafe, but secure, system is a capability machine which imposes no restrictions on capability

passing; this permits processes validly to acquire capabilities (i.e., rights) for objects without permission from the objects' owners. In contrast, a system that allows no sharing of objects at all would be safe, but of little interest where shared access to data bases is important. We are less interested in whether enforcement mechanisms are secure or precise than we are in whether the access control policy and its enforcement mechanism are safe.

Harrison, Ruzzo, and Ullman [HRU76] showed that it is undecidable whether a given, arbitrary protection system is safe. In their model, the safety question is formulated by asking "Can a specific right r be placed in the access matrix at a specific position (d,x)?". Should the answer to this question be "yes", the system is unsafe since the right r to x can be "leaked" to d. The generality of their result is underscored by the simplicity of their model of a protection system. Their model allows only a few primitive commands. Each command verifies that certain generic rights are present in the access matrix and, if so, performs one or more elementrary operations that may alter the state of the access matrix. Elementary operations allow creating or deleting objects or domains, and placing generic rights in, or removing them from, the access matrix. The undecidability of the safety question was established by showing how to transform the halting problem for an arbitrary Turing machine into a safety question; the construction shows how the

sequence of configurations of an access control matrix can simulate the configurations of a Turing machine, with the generic right of interest being placed in the access matrix position of interest if and only if the Turing machine halts. We state without proof their major results.

Theorem 2.1. The set of safe protection systems is not recursive.

Theorem 2.2. The set of unsafe protection systems is recursively enumerable.

Theorem 2.3. When no new objects or domains of access can be created, the set of safe protection systems is recursive and its decision problem is complete in polynomial space.

The first theorem is equivalent to saying that it is undecidable whether a given protection system is safe. The second theorem states that we can generate a list of all unsafe systems; this could be done by systematically enumerating all protection systems and all sequences of commands in each system, outputting the description of any system for which there is a sequence of commands causing a leak. We cannot, however, also enumerate all safe systems, for a set is recursive if and only if both it and its complement are recursively enumerable. The third theorem states that the safety question is decidable for systems of bounded size, but that any algorithm which decides safety can be expected to require enormous amounts of time.[1]

Harrison, Ruzzo, and Ullman also considered a highly constrained class of systems permitting only "mono-operational" commands which perform at most one elementary operation. This unrealistically severe constraint only improves matters slightly.

Theorem 2.4. The set of safe mono-operational systems is recursive; however, its decision problem is NP complete.(2)

In other words, while the safety of a mono-operational system is decidable, proving its safety can be expected to require exponential time in the worst case. And, though finding a proof of unsafety is NP hard, such a proof is short, amounting simply to a command sequence which generates a leak (and whose length is bounded by a polynomial function of the size of the system).

3. Theories for general protection systems

As a prelude to theories of protection we review the basic concepts of theorem-proving systems. A formal language L is a recursive subset of the set of all possible strings over a given finite alphabet; the members of L are called sentences.

A deductive theory T over a formal language L consists of a set A of axioms, where $A \subseteq L$, and a finite set of rules of inference, which are recursive relations over L. The set of theorems of T is defined inductively by:

(a) if t is an axiom (i.e., if $t \in A$), then t is a theorem of T; and

(b) if $t_1, \ldots, t_k$ are theorems of T and $\langle t_1,\ldots,t_k,t\rangle \in R$ for some rule of inference R, then t is a theorem of T.

Thus every theorem t of T has a proof which is a finite sequence $\langle t_1,\ldots,t_n\rangle$ of sentences such that $t = t_n$ and each t_i is either an axiom or follows from some subset of $t_1, \ldots, t_{i-1}$ by a rule of inference. We write $T \vdash t$ to indicate that t is a theorem of T or is provable in T.

Two theories T and T' are said to be equivalent if they have the same set of theorems. Equivalent theories need not have the same axioms or rules of inference.

A theory T is recursively axiomatizable if it has (or is equivalent to a theory with) a recursive set of axioms. The set of theorems of any recursively axiomatizable theory is recursively enumerable: we can generate effectively all finite sequences of sentences, check each to see if it is a proof, and enter in the enumeration the final sentence of any sequence which is a proof.

A theory T is decidable if its theorems form a recursive set.

Since the set of safe protection systems is not recursively enumerable, it cannot be the set of theorems of a recursively axiomatizable theory. This means that the set of all safe

protection systems cannot be generated effectively by rules of inference from a finite (or even recursive) set of safe systems. (Note that this does not rule out the possibility of effectively generating smaller, but still interesting classes of safe systems.) This observation can be refined, as we proceed to do, to establish further limitations on any recursively axiomatizable theory of protection.

Definition 3.1. A representation of safety over a formal language L is an effective map $p \rightarrow t_p$ from protection systems to sentences of L.

We wish to interpret t_p as a statement of the safety of the protection system p.

Definition 3.2. A theory T is adequate for proving safety if and only if there is a representation $p \rightarrow t_p$ of safety such that

$$T \vdash t_p \quad \text{if and only if} \quad p \text{ is safe.}$$

Analogs of the classical Church and Gödel theorems for the undecidability and incompleteness of formal theories of arithmetic follow for formal theories of protection systems.

Theorem 3.1. Any theory T adequate for proving safety must be undecidable.

This theorem follows from Theorem 2.1 by noting that, were

there an adequate decidable T, we could decide whether or not a protection system p were safe by checking whether or not $T \vdash t_p$.

Theorem 3.2. There is no recursively axiomatizable theory T which is adequate for proving safety.

This theorem follows from Theorems 2.1 and 2.2. If T were adequate and recursively axiomatizable, we could decide the safety of p by enumerating simultaneously the theorems of T and the set of unsafe systems; eventually, either t_p will appear in the list of theorems or p will appear in the list of unsafe systems, enabling us to decide the safety of p.

Theorem 3.2 shows that, given any recursively axiomatizable theory T and any representation $p \rightarrow t_p$ of safety, there is some protection system whose safety either is established incorrectly by T or is not established when it should be. This result in itself is of limited interest for two reasons: it is not constructive (i.e., it does not show how to find such a p); and, in practice, we may be willing to settle for inadequate theories as long as they are sound, that is, as long as they do not err by falsely establishing the safety of unsafe systems. The next theorem overcomes the first limitation, showing how to construct a protection system p which is unsafe if and only if $T \vdash t_p$; the idea is to design the commands of p so that they can simulate a Turing machine that "hunts" for a proof of the

safety of p; if and when a sequence of commands finds such a proof, it generates a leak. If the theory T is sound, then such a protection system p must be safe but its safety cannot be provable in T.

Definition 3.3. A theory T together with a representation $p \rightarrow t_p$ of safety is sound if and only if p is safe whenever $T \vdash t_p$.

Theorem 3.3. Given any recursively axiomatizable theory T and any representation of safety in T, one can construct a protection system p for which $T \vdash t_p$ if and only if p is unsafe. Furthermore, if T is sound, then p must be safe, but its safety is not provable in T.

Proof. The proof of Theorem 2 in [HRU76] shows how to define, given an indexing $\{M_i\}$ of Turing machines and an indexing $\{p_i\}$ of protection systems, a recursive function f such that

(a) M_i halts $\Leftrightarrow$ $p_{f(i)}$ is unsafe.

Since T is recursively axiomatizable and the map $p \rightarrow t_p$ is computable, there is a recursive function g such that

(b) $T \vdash t_{p_i}$ $\Leftrightarrow$ $M_{g(i)}$ halts;

the Turing machine $M_{g(i)}$ simply enumerates all theorems of T, halting if t_{p_i} is found. By the Recursion Theorem [Rogers, Section 11.2], one can find effectively an index j such that

(c) M_j halts $\Leftrightarrow$ $M_{g(f(j))}$ halts.

Combining (a), (b), and (c), and letting $p = p_{f(j)}$, we get

(d) $T \vdash t_p$ <=> $M_{g(f(j))}$ halts
<=> M_j halts
<=> $p = p_{f(j)}$ is unsafe,

as was to be shown.

Now suppose that T is sound. Then t_p cannot be a theorem of T lest p be simultaneously safe by soundness and unsafe by (d). Hence, $T \nvdash t_p$, and p is safe by (d). []

The unprovability of the safety of a protection system p in a given sound theory T does not imply p's safety is unprovable in every theory. We can, for example, augment T by adding t_p to its axioms. However, Theorem 3.3 states that there will exist another safe p' whose safety is unprovable in the new theory T'. In other words, this abstract view shows that systems for proving safety are necessarily incomplete: no single effective deduction system can be used to settle all questions of safety.

The process of extending protection theories to encompass systems not provably safe in previous theories creates a progression of ever stronger deductive theories. With the stronger theories, proofs of safety can be shortened by unbounded amounts relative to the weaker theories. (Gödel [Gödel36] discussed this phenomenon in logic and Blum [Blum67] discussed similar phenomena in computational complexity.)

Theorems 3.2 and 3.3 force us to settle for attempting to construct sound, but necessarily inadequate, theories of protection. What goals might we seek to achieve in constructing such a theory T? At the least, we would want T to be nontrivial; theories that were sound because they had no theorems would be singularly uninteresting. We might also hope that the systems whose safety was provable in T, when added to the recursively enumerable set of unsafe systems, would form a recursive set. If this were so, then we could at least determine whether T was of any use in attempting to establish the safety or unsafety of a particular protection system p before beginning a search for a proof or disproof of p's safety. The next theorem shows that this hope cannot be fulfilled.

Theorem 3.4. Given any recursively axiomatizable theory T and any sound representation of safety in T, the set

$$X = \{ p: T \vdash t_p \text{ or } p \text{ unsafe} \}$$

is not recursive.

Proof. If X were recursive, then the safety of a protection system p could be decided as follows. First, we check to see if p is in X. If it is not, then it must be safe. If it is, then we enumerate simultaneously the theorems of T and the unsafe systems, stopping when we eventually find either a proof of p's safety or the fact that p is unsafe. []

4. Theories for protection systems of bounded size

Real systems are finite. Some systems are designed for a fixed maximum number of users; others are designed to be extendable to any number of users, but at any given time are configured to handle only a given finite number.

If we consider finite systems in which the number of objects and domains of access cannot grow beyond the number present in the initial configuration, then the safety question becomes decidable, although any decision procedure is likely to require enormous amounts of time (cf. Theorem 2.3). This doubtless rules out practical mechanical safety tests for these systems. However, this does not rule out successful safety tests constructed by hand: ingenious or lucky people might always be able to find proofs faster than any mechanical method. We show now that even this hope is ill-founded.

Although we can always obtain shorter safety proofs by choosing a proof system in which the rules of inference are more complicated, it makes little sense to employ proof systems whose rules are so complex that it is difficult to decide whether an alleged proof is valid. We shall regard a logical system as "reasonable" if we can decide whether a given string of symbols constitutes a proof in the system in time which is a polynomial function of the string's length. Practical logical systems are reasonable by this definition. We show now that, corresponding

to any reasonable proof system, there are protection systems which are bounded in size, but whose safety proofs or disproofs cannot be expected to have lengths bounded by polynomial functions of the size of the protection systems.

Theorem 4.1. For the class of protection systems in which the number of objects and domains of access is bounded, safety (or unsafety) is polynomial verifiable by some reasonable logical system if and only if PSPACE = NP, that is, if and only if any problem solvable in polynomial space is solvable in nondeterministic polynomial time.(3)

Proof. (<=) By Theorem 2.3, the safety and unsafety problems for systems of bounded size are both in PSPACE. Hence, if PSPACE = NP, then there would be NP-time Turing machines to decide both safety and unsafety. Given such machines, we could define a reasonable logical system in which safety and unsafety were polynomial verifiable: the "axioms" would correspond to the initial configurations of the Turing machines and the "rules of inference" to the transition tables for the machines.

(=>) Also by Theorem 2.3, any problem in PSPACE is reducible to a question concerning the safety (or unsafety) of a protection system whose size is bounded by a polynomial function of the size of the original problem. Now if the safety (or unsafety) of protection systems with bounded size were polynomial verifiable, we could decide safety (or unsafety) in NP-time by

first "guessing" a proof and then verifying that it was a proof (performing both tasks in polynomial time). By Theorem 2.3, we could then solve any problem in PSPACE in NP-time, showing that PSPACE = NP. []

Since the above result applies equally to proofs of safety and unsafety, one must expect that there are systems for which it will be just as difficult and costly to penetrate the system as to prove that it can (or cannot) be done. In mono-operational systems, however, the situation is quite different.

Theorem 4.2. The unsafety of mono-operational systems is polynomial verifiable.

Proof. This result follows from Theorem 2.4, which shows that the unsafety question for mono-operational systems is solvable in NP-time. Alternatively, we simply observe that to demonstrate unsafety, one need only exhibit a command sequence leading to a leak. From [HRU76] we know that there are short unsafe command sequences if any exist at all: an upper bound on the length of such sequences is $g(m+1)(n+1)$, where g is the number of generic rights, m the number of domains of access, and n the number of objects. Thus an unsafe sequence (if it exists) has a length bounded by a simple polynomial function of the system size. []

By Theorems 2.3 and 4.2, proofs of unsafety for mono-operational systems are short, but the time to find the proofs cannot be guaranteed to be short; at the worst we might have to enumerate each of the sequences of length $g(m+1)(n+1)$ that could produce a leak. However, while proofs of unsafety are short for mono-operational systems, proofs of safety are not.

Theorem 4.3. For mono-operational systems, safety is polynomial verifiable if and only if NP is closed under complement.[4]

Proof. If NP were closed under complement, then safety would be in NP because unsafety is in NP by Theorem 4.2. Thus there would be a nondeterministic Turing machine for checking safety in polynomial time, which would demonstrate that safety is polynomial verifiable.

Conversely, suppose that safety were polynomial verifiable. We could then construct a nondeterministic Turing machine which would guess a proof of safety and then check it in polynomial time; hence safety would be in NP. But unsafety is in NP by Theorem 4.2, and if any NP complete problem has its complement in NP, then NP is closed under complement [Kar75]. []

These results imply that system penetrators have a slight advantage when challenging mono-operations systems: any system that can be penetrated has a short command sequence for doing so. However, it may still take enormous amounts of time to find such

sequences, as no systematic method of finding an unsafe command sequence in polynomial-bounded time is likely to be found.

5. Towards safe protection systems

Our primary interest in studying the provability of safety questions for protection systems is metaphysical: understanding the fundamental limits of our abilities to build provably secure computer systems. The results confirm widely-held suspicions that there is no hope of proving the security of arbitrary systems in any systematic or economical fashion. Moreover, they suggest that practical restrictions leading to tractable proofs of safety may be hard to find.

Our results for unbounded systems began from earlier results that showed how to encode an arbitrary Turing machine's configurations into the states of an access control matrix, leading thereby to the conclusion that proving a system safe is equivalent to solving the halting problem. We extended this result to demonstrate that any fixed logical system is inadequate for establishing the safety of all protection systems: there will always be a protection system whose safety is neither provable nor disprovable. We outlined the construction of such a system.

When attention is shifted to the protection systems of bounded size we are likely to encounter in practice, the earlier

questions become decidable, but intractably so. The set of safe protection systems of bounded size is decidable now, but its decision problem is complete among problems solvable in polynomial space, which means that any decision procedure is likely to require an amount of time at least exponential in the size of the protection system. It also means that safety proofs or disproofs are likely to be exponentially long (again in the size of the protection system).

As an illustration of restrictions within which proofs could be tractable, we considered protection systems with mono-operational commands. For these systems, the length of a proof of unsafety can be bounded by the product of the dimensions of the protection system; but proofs of safety are still long. Moreover, though a proof of unsafety may be short, finding it is likely to require at least exponential time.

In another study, Harrison and Ruzzo [HaR76] considered other protection systems whose commands are "monotone", i.e. whose commands never delete any domains, objects, or generic rights from the system. Since monotone systems are powerful enough to simulate Turing machines, all the above results apply to them as well.

These results support skepticism toward proving systems safe, for there is no single, systematic, general approach to establishing the safety or unsafety of arbitrary protection

systems. Hence we are forced to deal with approaches that are less general, or that attack the problem from a different point of view. Following are several possible approaches to the problem.

(1) Despite the undecidability and intractability results, we can still try to prove particular protection systems safe. After all, the incompleteness, undecidability, and intractability results in number theory have not stopped mathematicians from trying to prove interesting theorems, and so our results for protection systems should not stop computer scientists from trying to prove the safety of interesting protection systems. Our results merely stand as a warning that proving systems safe is not likely to be easy.

(2) In this respect, it would be be useful to develop techniques for proving the safety of sufficiently simple protection systems. For example, Lipton and Snyder [LiS77] investigated a class of systems which possesses a linear-time algorithm for deciding safety. On the other hand, the same authors [LiS78] studied another class of protection systems whose safety question is equivalent to a problem about vector addition systems (which seems to require at least exponential time even though it is decidable). Further study of such classes should prove fruitful.

(3) Rather than trying to guarantee the safety of a

protection system, which might be expensive, we might instead seek to give shorter demonstrations that the system is "probably safe" or "safe beyond a reasonable doubt". One possible approach might be to construct theories of protection which occasionally, though with very low probability, produced a "proof" that an unsafe system was safe.

(4) Finally, recognizing the well-known fact that certain access paths, known as "covert channels", may well be too expensive to eliminate [Lip75], we might concentrate on trying to prove that any way of compromising a given protection system must likewise be too expensive. Given this approach, the question of central importance would not be whether we could prove that a system is safe, but whether we could prove that finding a breach of security is, say, NP-hard or even harder.

Footnotes

(1) More precisely, it was shown that the safety problem for these systems can be solved in polynomial time (time proportional to a polynomial function of the length of the description of the system) if and only if PSPACE = P, i.e. if and only if any problem which can be solved in polynomial space can also be solved in polynomial time. Although the relationship between time and space is not well understood, it is believed that PSPACE $\neq$ P; in fact, it is believed that exponential time is required for such problems.

(2) The NP complete problems consititute a large class of problems, one of which is the well-known traveling salesman problem; if any one of these problems could be solved in polynomial time, then every problem in the class could be solved in polynomial time. However, it is widely believed that exponential time is required to solve these problems. See [AHU75] for a more thorough treatment of NP complete problems.

(3) PSPACE is the class of all problems which can be solved in polynomial space. It is known that any problem which can be solved nondeterministically in polynomial space is in PSPACE, but it is widely believed that PSPACE $\neq$ NP.

(4) It is considered unlikely that NP is closed under complement.

References

AHU75 Aho, A. V., Hopcroft, J. E., and Ullman, J. D., The Design and Analysis of Computer Algorithms, Addison Wesley Publishing Co., Reading, Mass., 1975.

Blu67 Blum, M., "A machine independent theory of the complexity of recursive functions", J. ACM 14 (1967), 322-336.

Göd31 Gödel, K., "Über formal unentscheide Sätze der Principia Mathematica und verwandter Systeme", Monatshefte für Mathematik und Physik 38 (1931), 173-198.

Göd36 Gödel, K., "Über die Länge von Beweisen", Ergebnisse eines mathematisches Kolloquium 4 (1936), 34-38.

HaR78 Harrison, M. A., and Ruzzo, W. L., "Monotonic protection systems", Foundations of Secure Computation, Academic Press, 1978.

HRU76 Harrison, M. A., Ruzzo, W. L., and Ullman, J. D., "On protection in operating systems", Comm. ACM 19 (1976), 461-471.

Kar75 Karp, R. M., "Lecture notes on automaton-based complexity theory", Computer Science Department, Univ. of California at Berkeley (1975).

Lip75 Lipner, S. B., "A comment on the confinement problem", Proc. 5th Symposium on Operating Systems Principles (Nov. 1975), 192-196.

LiS77 Lipton, R. J., and Snyder, L., "A linear time algorithm for deciding subject security", J. ACM 24 (1977), 455-464.

LiS78 Lipton, R. J., and Snyder, L., "On synchronization and security", Foundations of Secure Computation, Academic Press, 1978.

Rog67 Rogers, H. Jr., Theory of Recursive Functions and Effective Computability, McGraw-Hill Book Co., New York, N. Y. (1967).