情報保護機構の構成に関する問題点

池田克夫
（筑波大学学術情報処理センター）

1. 情報保護機構

情報保護の必要性については古くから論じられている。個人的に利用する計算機システムにおいてもシステムの信頼性やエラーによる被害を局限するために要求されるが、公共情報処理システムにおいては、システムの公平なサービスの保証、利用者の利益やプライバシーの保護は不可欠である。これらの要求は共有されるシステムにおいて各利用者を"分離"したり、それらの影響する範囲を"閉じ込め"たりすることによって満たすことができる。

公共情報処理システムのもう一つの役割は情報やその他の資源の共有にある。システムにおいて共通に利用されるオペレーティング・システム、各種の言語処理プログラム，ユーティリティ・プログラム，各種のデータベースをはじめとして、システムあるいは特定の利用者が利用料金を徴集して利用させる"販売プログラム"や"データベース"などは適当なアクセス制御の下に共用される。これらの共用は利用者あるいは提供者の過失あるいは悪意のあるなしに拘らず、双方にとって安全でなければならないし、共用によるデータベースなどにおいては、アクセス制御に加えて、同時アクセスにおけるデータ更新によってデータベースの全体性を欠くことのないような配慮も必要である。

Saltzer の定義によると(1)：

プライバシとは個人あるいは1つの組織が、その個人あるいは組織の情報を、いかに、いつ、だれに知らせてよいかを決定する能力をいう。

保全性（セキュリティ）とは情報システムに関して、格納された情報をだれが利用したり変更したりできるかを制御する機構と技術をいう。

保護（プロテクション）とは 1. 広義には保全性，2. 狭義にはプログラム実行による、計算機内に格納された情報の参照を制御する機構と技法をいう。

本稿においては、情報保護機構の構成に関するサーベイを行い、その問題点を挙げてみようとするものである。

2. セキュリティの侵害

セキュリティの侵害は、三つに類形化できる。(1)(2)(3)

1. 権限のないのに情報を参照すること。

権限のないのに情報を読み，利用したり、コピーしたり、あるいは情報のパターンを観察して推測することを含む。統計データの場合に母集団があまりに小さいと、特定の個別データまでわかってしまうことがあるが、この例もこのような侵害と考えられる。コピーの手段としては、情報を無断で他のファイルにコピーしたり，メッセージの形で他に送るようなことの他に、同期のロックを用いて"符号化"することなども含まれる。

2. 権限のないのに情報を変更したり破壊すること。

この場合加害者は情報の内容を見られなくてもよい。

3. 権限のないのに正当な利用者の利用を防害したり、利用しにくくすること。

加害者が読み、書き変更できない場合でも、システム・クラッシュを引起したり、スケジューリング、アルゴリズムを変更したりしてシステムの利用を防害すること。または、正しく機能しないように変更すること。

3. 情報の保護と共有のレベル

保護のレベルには低級なものから高級なものへと次のものが考えられる。(1)

1. 無保護のシステム。
2. 全てか無かのシステム。

従来のプロブレムモードとスーパバイザ・モードによる2層のシステムはこのクラスに属する。スーパバイザ・モードになると全ての権限を得ることができる。

3. 情報の制御された共有を認めるシステム。CTSS, DEC, ADEPT, TENEX等。
4. 利用者がアクセス制御の方法を任意に規定することのできるシステム。
   Multics, CAL, UNIX, Bcc-500, CAP, HYDRA 等。
5. 情報が参照された後でその情報の保護を管理するシステム。

参照された情報に対する保護は計算機の中にある場合と外にある場合とでは、同一であるべきであろが、実際には、計算機の外に出た場合には同等の保護が及ばない。

情報共有のレベルには低級なものから高級なものへと次のものが考えられる。(2)

1. 共有を認めないシステム。
2. 情報の写しを共有させるシステム。
3. 情報の原本を共有させるシステム。
4. サブシステムを共有させるシステム。
5. 相互に疑わしいサブシステムの協同を認めるシステム。
6. メモリレスを保証するシステム。
7. 正しいことを証明したシステム。

今日、実現されているのはレベル4までであり、レベル5以上のものは提案の域を出ないように思われる。

4. 情報保護機構の設計の原則と評価の尺度

情報保護の機構は、権限を認められているものの参照を許し、権限のないものに対しては、これを全て排除するという消極的な動機に基づいているために、現在のところ完全な設計方法が明らかにされていない。

ここでは保護機構の設計に関して有用と考えられる原則を示す。(1)

1. 経済性　経済性は全てに適用される原則である。経済性を無視した機構は利用されず、従って、システムの保護が行われないという結果となる。
2. フェイル・セーフ　基本的には排除より許可という決定法に基づくべきである。許可の機構が故障した場合には、アクセスできないが情報そのものは保護されている。
3. 参照の都度チェックする。アクセス権の変化にも速応することができる。
4. 設計の公開　加害者が知らないから安全になるとは言えない。侵入口が残っていることをかくすために設計を公開しないとしたら、これは安全ではない。

5. 権限を分割して、多重に保護を行って安全性を高める。

6. 最小限の権限のみを与える。

7. 各利用者にとって共通の機能部分を最小にする。共通部分は、各利用者を"分離"するという原則に反する。

8. 心理的に受入れやすい機構にする。これにより、利用者の保護機構の利用を促進する。

情報保護機構の評価の尺度としては次の4点が考えられる。[11]

1. 機能　保護機構が要求される機能を有することが第1に必要である。

2. 経済性　利用した機能に応じた経費となること、かつ、経済的であること。

3. 単純性　保護機構が単純であること。間違った利用によって却って不安全な使用法となる危険があり、利用者が正しいという確信を得ることもできない。

4. プログラムの一般性　保護とプログラムの論理とは本来独立なものである。従って、保護機構を利用するためにプログラムを変更しなければならないというのはナンセンスである。

5. 情報保護機構の構成

情報処理システムの保全のための技術は極めて広い範囲にわたっている。情報保護機構としては計算機内の機構に加えて、計算機室の施錠や利用者、オペレータの管理策など、もろもろの天災・人災に対する防護策、端末との通信線を介する情報伝送における暗号化の技術をも実際には考慮しなければならない。一般的な議論は他にゆずることにして、ここでは狭義の保護機構として、計算機内の機構について論じることにする。

5.1 情報保護機構の構成要素

保護機構は3つの要素によって構成される。[8]第1の要素は、アクセスが制御されるべき目的物（客体）の集合である。主記憶のページ、ファイル、プログラム、2次記憶装置などは客体の例である。客体が情報である場合、通常、セグメントと呼ばれる。[7]セグメントは情報の論理的なーかたまりであり、情報保護のための単位として用いられることが多い。各客体には、これが作成あるいは定義された時点でオペレーティングシステムによって捏造不能な(unforgeable)一義的な識別名がつけられている。これは作成時点での時計の値などを用いることが一般に行われる。

第2の要素は、客体にアクセスする主体であって、これのアクセスの仕方が制御の対象とされる。主体は、プロセスとその実行状態の組によって規定される。実行状態の例はプロブレム・モードあるいはスーパバイザ・モードの別、またはMulticsのリングなどである。実行状態は具体的には、領域(domain)、リング、環境(sphere)などと呼ばれることもある。主体は保護されるべき対象でもあるので、主体であると同時に客体でもあり、一義的な識別名がつけられる。

第3の要素は、主体による客体のアクセスを制御する規則である。この規則は保護システムのかなめである。この規則の定め方によって保護システムのレベルが決定される。

問題となるのは、1)主体の保護に関する実行状態の表現とシステムの保護の規

則を表現すること，2)主体のアクセスを規則に従って制御すること．3)システムの保護の規則を必要に応じて変更することである．

1)はアクセス行列により表現することができる．行列の要素 A(S,X) は主体Sの客体Xに対するアクセス権を示す．客体には種々の形(type)があり，各々の形に対して，その形に要求されたアクセスが認められるか否かを決定するモニターを用意することによって，2)の問題を解決することができる．モニターの例はファイルに対するファイル・システムや命令実行のためのハードウェアをあげることができる．3)の問題はアクセス行列を変更するためのモニターを用意し，"owner"と"control"，および"copy"というアクセス権とアクセス行列用のモニターが使用する3つの規則[3]を用意することによって実現することができる．

主体を創成することは，創成の主体にownerアクセスを，創られた主体にそれ自体に対するcontrolアクセスを与えることによって行われる．主体の抹消はその主体のownerである主体により行うことができる．主体の実行状態の変更は，主体が遷移すべき先の主体にスイッチするためのアクセス権を与えることにより行うことができる．

5.2 アクセス行列のインプレメント

一つのシステムにおいて，アクセス行列を規定する主体も客体も非常に個数が大きく，アクセス行列は当然のことながらスパースな行列となり，このままでは実現が困難である．このため，

1．アクセス権を主体の権限として表現する．即ち，アクセス行列の行を抽出したケーパビリティ・リスト(C-リスト)の表現をとる．

2．アクセス権を客体に対する許可として表現する．即ち，アクセス行列の列を抽出したアクセス制御表(ACL)の表現をとる．

3．折衷的な方法をとる．

ことが行われている．

1は，主体としてのプロセスの権限を陽に表現している点で，直接的な表現であり，計算機システムの主記憶中の情報をアクセスする場合は極めて能率よく実現できる可能性がある．CPUが情報参照のためにメモリ・アドレスを指定することは，一つの権限を示していると考えることができる．この意味からこの権限をケーパビリティ[7]と呼ぶ．ケーパビリティ・ベースド・アドレッシング[4]という概念はこれを明確にしたものである．ケーパビリティは"切符"(ticket)と考えることができる．1の方式は，客体に対する権限の取り消し(revocation)が困難である．即ち，ある客体に対する権限を利用しようとするとき，その権限は，一般に複数の主体に分与されていたり，別の主体に移動されていることもあり，これを完全に把握することは特別の工夫を必要とする．従って，この方法をファイル・システムに置かれた客体に対して適用することは，困難が予想される．

2の客体に対する許可としてのACLは，コンピューター・ユーティリティのファイル・システムのように対象が極めて多い場合に有効であり，いったん与えたアクセス権の取り消しも容易に行うことができる．(リスト方式)しかし，この方法は，"間接的"である．即ち，ある主体が一つの客体を参照したいと欲するとき，ACLをスキャンしてその権限をチェックすることが必要である．プログラムの実行において，常時この方法に頼ることは実行の能率や，記憶容量の点から適当でない．[5] 従って，ある客体が最初にある主体によって参照されると

き、ACLがチェックされ、これをいったんアクティブにしたならば、C-リストに移すのが実際的である。この際、C-リストに移された客体に対する権限は、ポインタを用いて、論理的な結合を元のACLとの内で保って、権限の取り消しを容易にする必要がある。この方法はMulticsにおいて用いられている。[6]

5.3 領域の構成

情報保護の基本的な考え方は、最小限の権限のみを与えるという原則によって、一つの主体がアクセスできる客体の範囲を限定すること（隔離）、および、情報の流れを限定すること（閉じ込め）である。領域（domain）とは、プロセスの実行状態に応じて定まるアクセス権限（ケーパビリティ）の集合である。領域は、プロセスの実行状態を示す属性とも考えられることから、"領域を実行する"というような言い方がされることもある。

ケーパビリティは、客体の識別情報とアクセスの種類により示される。客体の識別情報は、1)客体の識別名や、2)客体の格納されているメモリ内の記憶番地として直接的に示されることもあるし、3)記憶番地を示す表のインディックスとして間接的に与えられることもある。

ケーパビリティを与えるC-リストは、ケーパビリティ・セグメントに格納される。セグメンテーションの環境で、ケーパビリティが記憶番地によって示される場合、ケーパビリティ・セグメントはデスクリプタ・セグメントとして実現することができる。セグメントは情報の論理的な一単位であるから、この単位にアクセス権をつけて、アドレス変換の都度アクセス権のチェックも行うようにすればよい。

ケーパビリティはセンシティブな情報であるから、その処理は当然、限定された方法でのみ許される。実際のインプレメントにおいては、ケーパビリティを処理するためのデータの流れと、一般のデータの流れを分離したり、命令の組を別にしたりすることが行われる。

ケーパビリティと一般データをセグメント単位で分ける方法に対し、メモリの各語に、どの種類（type）の情報かのタグをつける方式も提案されている。[10]Burroughsなどがその例である。全ての情報はタグによってその意味づけが明らかにされていて、CPUの命令は常にタグを調べ、それに見合った操作しか行わない。前者は技術の問題であり、後者は本質的な問題であるから、将来はタグ方式が本命であるとする考え方もある。

一つのプロセスにおいて互いに独立な領域を構成するためには、複数のC-リストを用意して、これを切替えることが必要である。領域の切替えは手続きの呼出し／復帰により行うのが一般的である。C-リストのインプレメントの方法としてSchroeder[14]はC-リストの配列を用いる方法を提案している。Fabry[4]は、C-リストの与え方のいくつかのクラスを示している。リング保護機構はリング番号が大きくなるとC-リストがリング番号の小さなもののサブセットとなるようにして構成されている。ソフトウェアによりリング保護機構が実現される場合には、各リングにこのような関係のC-リストが用意される。ハードウェアによる実現は1つのC-リストの解釈を限定する方法でインプレメントされている。

C-リストのインプレメントの別の方法として、筆者は複数のC-リストを用いる方式について提案している。[15]ケーパビリティには、プロセスの個有の権限とみなされるもの、実行しているプログラム自身が個有に有する権限、手続きの呼

出し／復帰に伴う引数を参照するための権限というように数種類のものが考えられる。これらは本質的には1つのC-リストのサブリストに相当するものであるが、実行状態の変化によってC-リストの全体を取替えるよりも、失通でないもののみを切替える方がケーパビリティの管理，時間，記憶域等の面からより好都合と考えられる。

6．操作システムとプログラミング言語

情報保護の機構がいくら完全に設計されていても、その上で実行される操作システム等のサブシステムが完全でなければ意味がない。これはソフトウェアを作成する側に全ての責任が課せられる訳であるが、前述の基本的な原則に忠実に設計し、明確に構造化されたプログラミングを行うこと、および以下に述べるプログラミング言語の面から保全性の欠陥が極力少なくなる手段を講じることが必要であろう。

情報保護はプログラム言語の問題でもある。[12][13] 前述のケーパビリティとは番地であるという解釈は、ALGOL 68の変数がレファレンスであるという解釈とも一致する。また高水準プログラミング言語においては目的プログラムの書き直しが一般に不可能であってすでに保護ができている。また、ブロック構造を有している言語では変数にスコープがあって、実行状態によって領域がはっきり規定されていることを考えるとプログラム言語の役割は重要である。

7．情報の流れの限定

情報の流れを限定すること（閉じ込め：*confinement*）によって情報の保護を行おうとする考え方がある。[14] この閉じ込めを実行するための法則は次のものである。

1．メモリレスであること：主記憶や2次記憶にプログラムやデータの痕跡を残さない。プログラム言語の面で言えば、*own*変数や*global*変数のないALGOL手続きはメモリレスであるといえる。

2．完全な隔離を行う：直接的、間接的SVCを含めて一切の手続き呼出しを禁止する。これは実際的ではない。そこでスーパバイザーを“信用”することとして、次の規則を設ける。

3．遷移則：呼出された手続きは閉じ込めが完全であること。

システムが供給する記憶スペースは、初期化することによって、プログラムやデータの痕跡を残さないようにすることができる。問題となるのは外部セグメントをアクセスする場合（ALGOLの*global*変数に対応する）や、その他の通信チャンネルを利用して情報をリークすることである。これに対して筆者は所有者の権限による書き込みの禁止と課金チャンネルの設置を提案している。[16]

8．事例研究

8.1 Multics[11][16] ファイル・システムに対してはアクセス制御表（ACL）を用いセグメントがアクティブになると、これとC-リストを移す。1972年以降ハードウェアによるリング保護機構を使用している。セグメントの共有の統計として、通常のセグメント12%，ディレクトリ57%等の値が報告されている。

8.2 UNIX[17] ファイルの保護として、ファイル毎に識別名とファイルの所有者自身および、ファイルの借用者に対する読み、書き、実行を示すフラグと、プログラムの実行時に借用者の権限で実行するのか、所有者の権限で実行するのか（set user ID と呼んでいる）を区別するフラグを用意している。

8.3 ADEPT-50[18] 米国政府および軍関係の機密情報を扱うシステム。利用者、端末、ジョブ、プログラム、ファイル等を対象にして、権限、カテゴリ、免許権限によって制御する。

8.4 CAP[19][20][21][22] 番地はケーパビリティ、セグメントの中の一つのケーパビリティを示す。ケーパビリティはグローバルな表を参照して目的セグメントをアクセスする。保護に関して、SEALDATA, SEALCAPABILITY, UNSEALDATA, UNSEALCAPABILITY, ALTER, MODECAP, GETARG, RESULTS 等の命令が用意されている。保護機構はハードウェアとファームウェアによりインプレメントされている。効率に関してはロード命令 2.1μs ストア命令 1.9μs に対して手続きの ENTER + RETURN が 0.24ms, ENTER + RETURN + ケーパビリティの引数 1ヶ + ケーパビリティの結果 1ヶ が 0.47ms と報告されている。

8.5 HYDRA[23] 手続き毎に局所的な領域を構成し、手続き呼出し／復帰の都度切替える。ケーパビリティ・リストにより情報保護を制御するケーパビリティは、呼び出し者に独立なものに加えて呼出し者に依存するものを用いることができるようにしている。

8.6 CAL[24] ケーパビリティ・ベースの保護機構を用いている。オペレーティング・システムは保護の層の考えに基づいて構成されている。ケーパビリティは MDT と称するグローバル表を参照してからファイルを参照する。

8.7 ACOS[25] 手続き毎に領域を切替える。1つの手続きが保有するケーパビリティはリンケージ・セグメントと称する表に登録される。この他にデスクリプタ・レジスタを複数個有していて、これにより必要なセグメントや手続き呼出しに伴う引数の参照を行う。

9. おわりに

計算機システムにおける情報保護機構の研究は 1960 年代の後半より始められているが、本格的に問題にされ出したのは TSS のように多数の利用者が一つのシステムを利用することが一般に行われるようになってからである。ケーパビリティ・ベースド・システムとしての観点は比較的新しく、これに基づく CAP システムに関する論文は SOSP 6 で本格的に発表されている。商用の計算機システムにおいても領域を手続き呼出しの都度 切替える設計を行っているものも出現し始めている。

計算機のセキュリティ対策は今後一層重要性を増し、より一層複雑な保護機構が要求されると思われるので、本稿においては、主として計算機内の情報保護機構の構成に関して、保護の基準、情報共有の基準、保護機構設計の原則と評価の尺度、保護機構の構成要素およびインプレメンテーションについてサーベイし、

問題点を示した。さらに幾つかの計算機システムにおける保護機構について簡単に紹介した。

参考文献

(1) Saltzer, Schroeder, IEEE Proc. 63, 1278-1308, 1975.

(2) Lampson, SOSP5, 18-24, 1974.

(3) Graham, Denning, AFIPS 40, 417-429, 1972.

(4) Fabry, CACM 17, 403-411, 1974.

(5) Lampson, AFIPS 35, 27-38, 1969.

(6) Organick, MIT Press, 1972.

(7) Dennis, Van Horn, CACM 9, 143-155, 1966.

(8) Lampson, CACM 16, 613-615, 1973.

(9) Evans, Leclerc, AFIPS 30, 23-30, 1967.

(10) Feustel, IEEE C-22, 7, 644-656, 1973.

(11) Schroeder, Saltzter, CACM 15, 157-170, 1972.

(12) Morris, Jr., CACM 16, 15-21, 1973.

(13) 和田, 情報処理 16, 1092-1098, 1975.

(14) Schroeder, MAC-TR-104, MIT, 1972.

(15) 池田, 情報処理学会計算機アーキテクチャ研資料, 27-2, 1-10, 1977.

(16) Montgomery, SOSP6, 85-90, 1977.

(17) Ritchie, Thompson, CACM 17, 365-375, 1974.

(18) Weissman, AFIPS 35, 119-133, 1969.

(19) Herbert, SIGOPS 12, 1, 24-28, 1978.

(20) Needham, Walker, SOSP6, 1-10, 1977.

(21) Cook, SIGOPS 12, 2, 26-28, 1978.

(22) Needham, SOSP6, 17-22, 1977.

(23) Wulf, Cohen et al., CACM 17, 337-345, 1974.

(24) Lampson, Sturgis, CACM 19, 251-265, 1976.

(25) 渡辺, 京大大型計算機センター研究セミナ報告 8, 26-40, 1977.