ソフトウェアセキュリティと
ユーザー管理

2010年6月30日

海谷 治彦

目次

- セキュリティー般について
- IdentificationとAuthentication
- アクセス コントロール
- セキュリティ レベル
- セキュリティ モデル
  - 誰がどんなデータをアクセスできるかを表現するための方法

今回の話はセキュリティ関係の授業と被ります.
加えてネタ本が英語なので日本語訳が所々わからん・・・

セキュリティとは

- 資産(assets)の保護(protection)についてのことを指す.
  - 資産と価値(value)が何かは所有者等に理解されているというのが大前提.
- 保護の度合い
  - Prevention 資産の損害を回避する.
  - Detection 資産の損害を検知する.
  - Reaction 資産もしくは資産の損害を復旧する.

コンピュータにおける資産とは？

- コンピュータの情報や資源である.
- 情報 vs データ
  - データの主観的な解釈が情報である.
  - 無論，データとして情報は記録されている.
  - しかし，データの中身を守ったからといって情報を守れるとは限らない.
    - データが存在するか否か自体，情報なので，これをアクセス権限の無いものがアクセスできる場合がある. (covert channel)
    - 秘密にしたデータ以外から，秘密にしている情報を推論できてしまう場合がある.
- 資源: CPU利用時間，ディスク容量，デバイス他

資産保護の視点

下記の3つが代表的な視点といわれる.

- Confidentiality 権限無いものに情報が漏れるのを防ぐこと.
  - 最も一般的に認識されている側面.
- Integrity 権限ないものの情報の改ざんを防ぐこと.
  - HPや口座データの不正書き換え等
- Availability 権限無いものが情報・資源を制することを防ぐこと.
  - 要は正規ユーザーは情報やサービスを得たい時に得られること. DoS攻撃はこれを破る典型例.

コンピュータ・セキュリティ

- コンピュータのユーザーによる権限を越えた行為(action)を実行することを回避したり検知したりすること.

  authorization, access controlの概念が重要となる.
- Authorization 権限委任
  - 誰にどんな行為を許す/許さないを規定したルールであるsecurity policyをもとに委任は行われる.
- Access Control アクセス制御

IdentificationとAuthentication

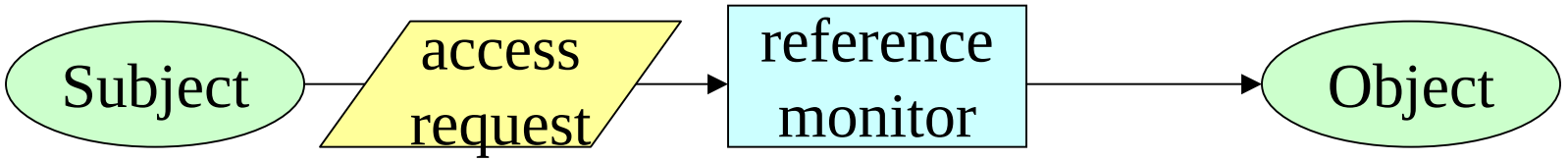
- Identificationはユーザーが自分が誰であるかをコンピュータに名乗ること.
- Authenticationは，そのIdentificationが正しいか検証すること.
- ユーザーの立場からは，例えば，ユーザー名とパスワードを入れるだけのことなので上記二つの区別はいまいちクリアじゃない.
- ユーザーのidentityをもとにアクセス制御が行われることが多いが，そうでない方が良い場合がある.
- 要はパスワード管理はちゃんとしようね，というオチですね.

Access Control

- ファイル等のコンピュータ資産を，誰がどやってアクセスして良いかを規定すること.
- これを規定するためには以下の二つを決める必要がある.
  - 考えているアクセス制御の方針を記述するための言語.
  - その制御を実施するためのメカニズム.
- Unix(Linux)のrwxの仕組みもこの例の一つ.

Subject, Object, Access Ops. 他

- 「能動的なsubjectが受動的なobjectに対して，あるaccess operationでアクセスしようとして，reference monitorがそのアクセスに許可を与えるか否かを決める．」
  - というのがアクセス制御の基本モデル．



Access Operation (AC)

- Unixにおけるファイルの読み書き(read and write)に相当するもの.
- しかし，コンピュータシステム(OSやDBMS)の種類によってどんなACが存在するか異なる.
  - 原始的なレベルではメモリやディスクの読み書きである.
  - しかしOODB(オブジェクト指向データベース)等の高い抽象度を持つシステムでは，個々のメソッド呼び出しがACとなる.

Access Mode

- 最も基本的なレベルでは，以下の2つのaccess modeが与えられる.
  - observe: objectの中身を見れる.
  - alter: objectの中身を変更できる.
- これら二つで基本的にどんなACも書けるはずだが，あまりに原始的なので，込み入った処理に関するACを記述しようとすると複雑になってしまう.

Access Rights

- 以下の4種類が一般的.
  1. execute ファイルに限らずデータ列をプログラムと見なしてCPUへ命令として送ること.
     - 無論, プログラムの体を成していなければ破綻するが.
  2. append writeと似ているが追加だけの場合, ファイル等の中身をobserveする必要がない.
  3. read
  4. write observe and alterに相当する.

上記の分類は後述のBLPに基づき, UNIXのそれとは若干異なる.

UNIXの場合

	file	directory
read	reading from a file	list directory contents
write	writing to a file	create or rename a file in the directory
execute	executing a program file	search the directory

特に，directory-writeに注意.
ファイルの作成消去はディレクトリのアクセス権で決まる！

また，directory-executeに注意.
dirが‘- - x’でも移動はできるという不思議.

NTFSの場合

- Windows NT, XPの採用するファイルシステムのアクセス制御.
  - read
  - write
  - execute
  - delete ファイルの消去はファイル自体の属性
  - change permission
  - change ownership

誰がアクセス権を決められるか？

この点に関しては以下の2つの流儀がある.

- Discretionary Access Control (DAC)
  - 資産の所有者がその資産のアクセス権を決定できる.
- Mandatory Access Control (MAC)
  - システム(OSやDBMS)が管理する資産全てのアクセス権を決定する.

Access Control Matrix

	bill.doc	edit.exe	fun.com
Alice	-	execute	execute, read
Bill	read, write	execute	execute, read, write

誰(Subject)が何(Object)をどう(operation)アクセスするかを規定する一般的な記法.

一般的にデカくなるので，実用的ではないかも.

Capabilities

- 誰(Subject)を基盤にアクセス権を記述する方式.
- 例
  - Allice: edit.exe: x; fun.com: x, r
  - Bill: bill.doc: r, w; edit.exe: x; fun.com: r, w, x

Access Control Lists (ACL)

- 何(Object)を基にアクセス権を記述する方式.
- 例
  - bill.doc Bill: r, w
  - edit.exe Alice: x; Bill: x
  - fun.com Alice: r, x; Bill: r, w, x
- UNIX(Linux)はこの方針を基本としている.
  - ただし，個々のユーザーではなく，本人，同一グループの人，その他の3つしか区別しないが.

セキュリティ レベル

- SubjectおよびObjectを順序付けすることで, アクセス制御を楽に定義することができる.
- このような順序付けのことをセキュリティ レベルという.
- Unix(Linux)の場合，Subjectの方に半順序(root ≧ それ以外のユーザー)というレベルがある.

半順序と全順序

- 以下の3つの性質を満たす≦を集合S上の半順序(partial order)と呼ぶ.
  - 任意の要素e ∈ S について, e ≦ e (反射律)
  - e≦f かつ f ≦eならば, e=f (反対象律)
  - e≦f かつ f≦gならば, e≦g (推移律)
- 加えて以下の性質を満たすと全順序(total order)と呼ぶ.
  - 任意の要素 e, f ∈Sについて, e≦fもしくは f≦eが成り立つ. (完全律)

セキュリティ モデル

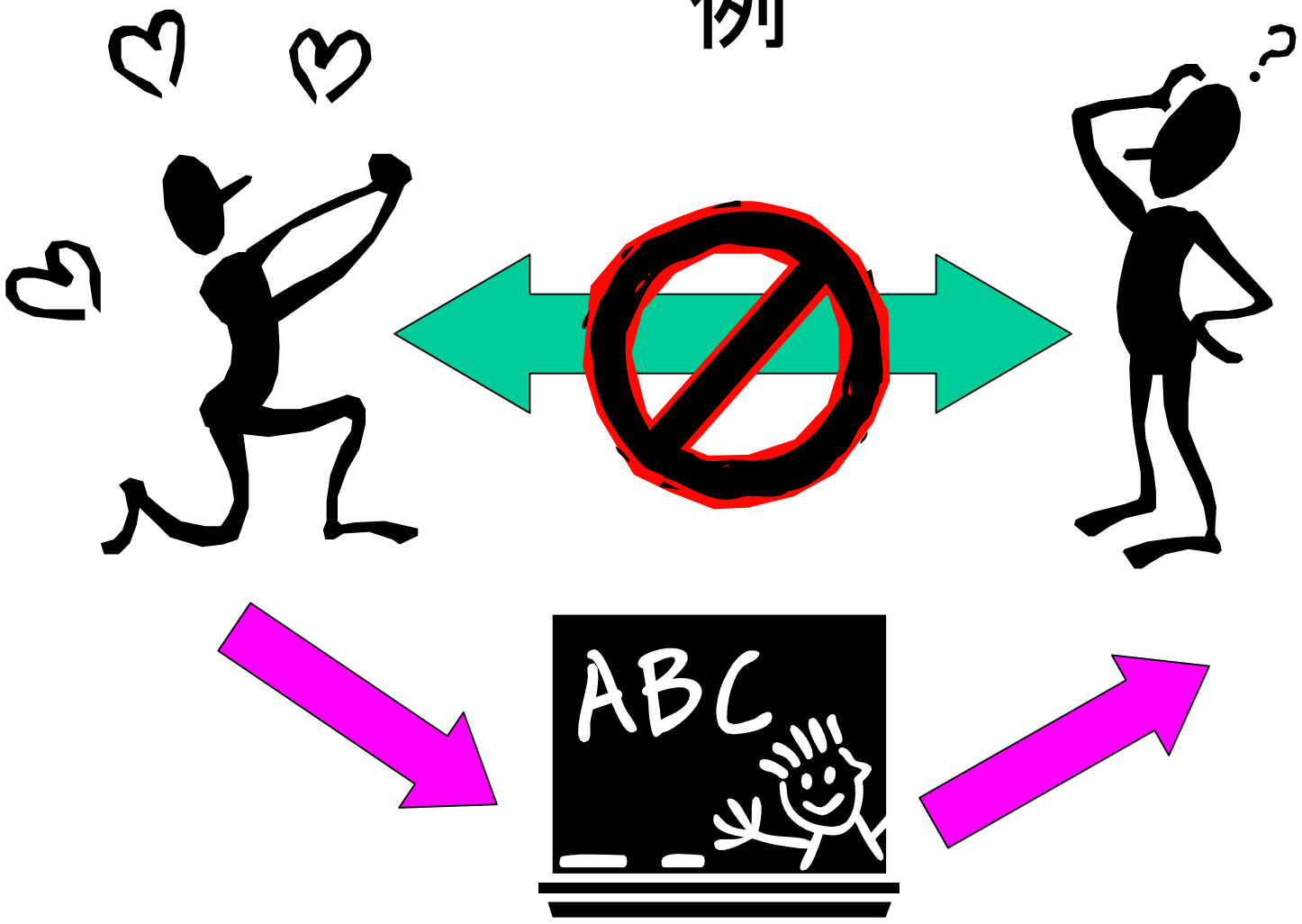
- 誰がどの資産にどうアクセスできるかを規定したルールがセキュリティポリシィである.
- セキュリティポリシィを記述するためには,
  - そのポリシィでどのようなSubjectやObject, operationを扱うか？
  - どのようにルールを記述すればよいか

  を決めなければならない.
- セキュリティモデルはポリシィ記述の枠組みを与える.
- 以降，有名なセキュリティモデルを紹介する.

Covert Channel

- 要は正規の通信経路ではなく裏技的な方法で情報伝達をする伝達路(channel)のこと.
- 無論，この伝達路を生かしておくと，Confidentiality (秘密を守ること)は保たれない.
- 教科書の定義では，この伝達路として共有資源(ファイルや伝言板等)を使う.

例



Covert Channel

Bell-LaPadula Model の伏線

- BLPの仮定
  - Subject(人等)は情報アクセスについてランク付けされている.
  - Object(データ)も同様にランク付けされている.
- BLPの目的
  - Subjectが自分の許可レベルより高いObjectを読むことができないようにすること.

Bell-LaPadula Model (BLP)

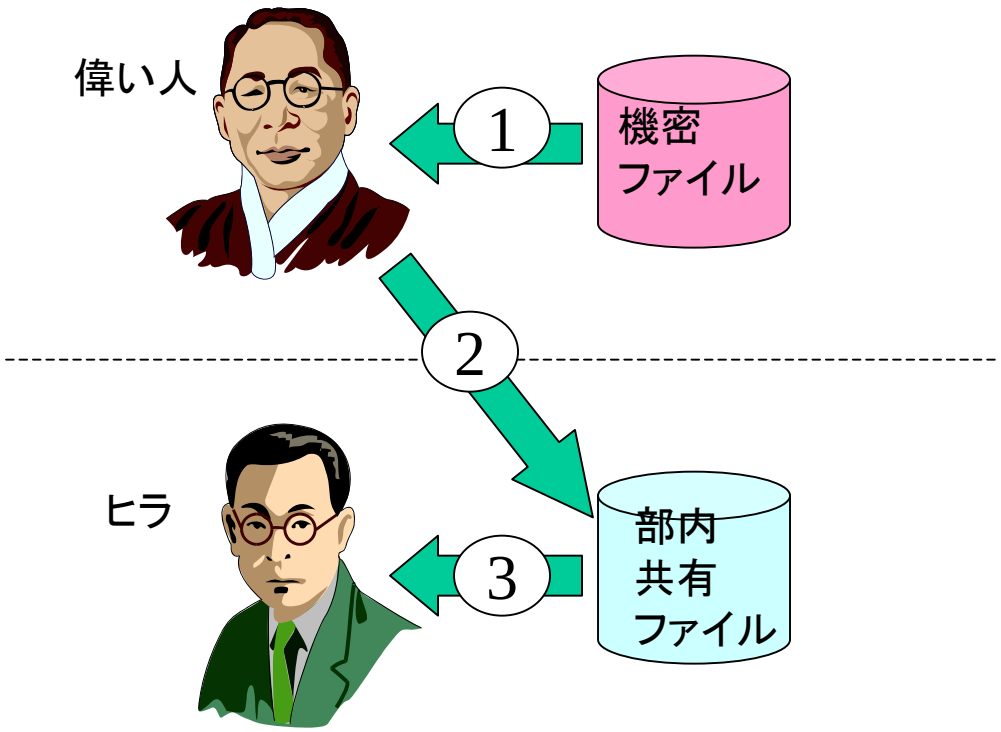
- 最も有名なモデルと言われ，confidentiality を表現できる.
- subjectの集合S, objectの集合O, operation(op.)の集合 A={execute, read, append, write}
- それぞれのsがoをどう操作してよいかは，アクセス行列Mで定義されている.
- SとOの要素は半順序≦でレベル付けされている.
- Sが持てる最大レベルと，現在のレベルという二つのレベルがある.
  - 無論，現在レベル≦最大レベル

BLPの性質1: ss-property

- 「oのレベル≦sの最高レベル の場合のみ, read, writeは実行できる. 」
- 実はこの性質だけでは, レベルの低いsが高いoをreadするのを完全に防止できない.
  - あるsは, 自分より低いレベルのoにwriteできるので, 高いレベルのsが手引きすれば, 低いレベルのsが高いレベルのoを読めてしまう.
    - 課長がヒラが見ちゃいけない書類をコピーして, ヒラに渡すようなもの.

ss-propertyの問題点

- 実はこの性質だけでは，レベルの低いsが高いoをreadするのを完全に防止できない.

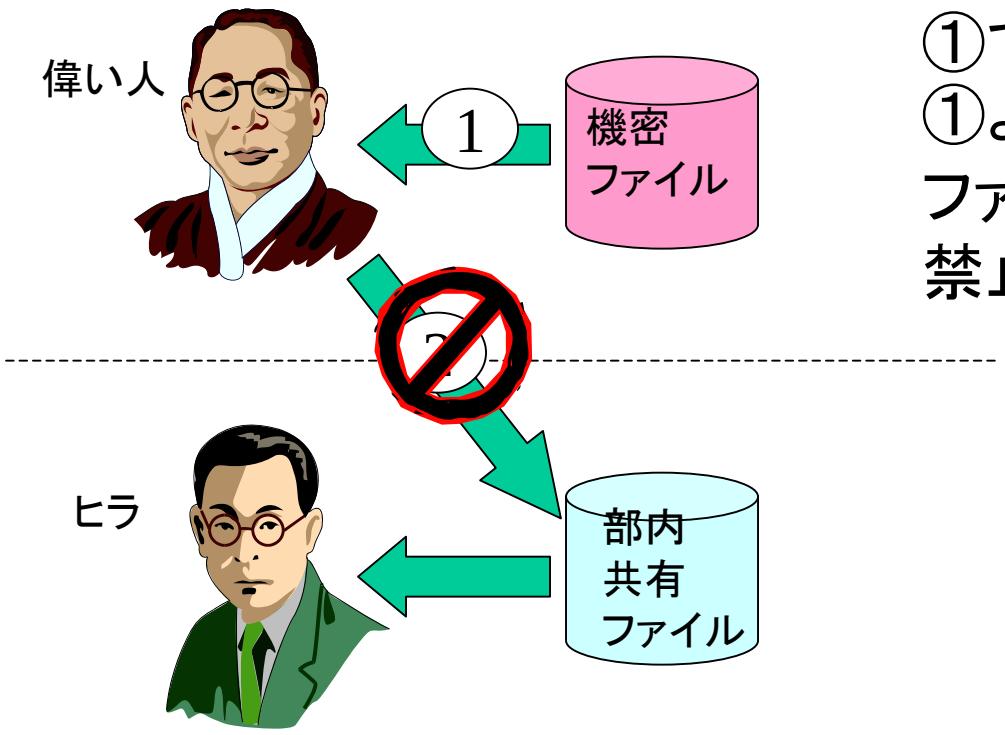


左記がおきてしまう.

BLPの性質2: star-property

- 「sの現在のレベル≦oのレベル の場合のみ，appendやwriteを実行できる．」
- 要は高い権限でreadして，それを低い権限でもreadできるoに書き込むことはできなくなる．
  - 高いレベルのsは低いレベルのs’に情報を送れなくなる．
  - read時のsのレベルが，write時のoのレベルを制約しているため．
- writeの場合，ssと合わせて，「sの現在のレベル≦oのレベル≦sの最高レベル」実行可能．
- appendの場合は，場合によってはsの最高レベルを超えるoにデータをappendできる．
  - BLPのappendはreadを伴わないため．

star-propertyの効用



star-propertyにより,
①で読んだ情報は,
①より下(秘密じゃない)
ファイルには書き込みが
禁止される.