

Security Models

CSH5 Chapter 9

“Mathematical Models of Computer Security”

Matt Bishop

1

Copyright © 2011 M. E. Kabay. All rights reserved.

Topics

- Why Models are Important
- Models & Security
- Models & Controls
- Classic Models
- Other Models
- Conclusion



2

Copyright © 2011 M. E. Kabay. All rights reserved.

Why Models are Important

- General description of system
- Definition of protection
- Conditions for protection
 - ❑ Mathematical models allow proof
 - ❑ (Assuming model properly implemented)
 - ✓ Assumptions are critically important
 - ✓ Must verify assumptions to validate applicability of model
- Chapter 9 presents different types of model
 - ❑ See next slides



3

Copyright © 2011 M. E. Kabay. All rights reserved.

Types of Model

- 1) Deciding if a system can be proved to be secure
 - 2) Describing how computer system applies controls
 - 3) Describing confidentiality & integrity
 - 4) Hybrid model mixes requirements
- Goals of this chapter
 - ❑ Study main models in information security
 - ✓ Meaning
 - ✓ Applicability
 - ❑ Become sensitive to assumptions underlying information assurance

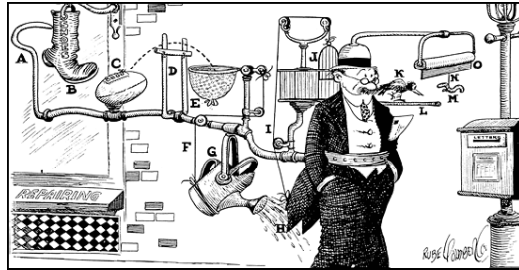


4

Copyright © 2011 M. E. Kabay. All rights reserved.

So Why are Models Important?

- Provide framework for
 - ❑ Analyzing systems
 - ❑ Focusing security efforts in right place
 - ✓ Validating assumptions; or
 - ✓ Ensuring assumptions are met in reality
- Mechanisms
 - ❑ Technical
 - ❑ Procedural
 - ❑ Quality of mechanisms determines security of system
- Overall: *model provides basis for confidence in possibility of effective security*



5

Copyright © 2011 M. E. Kabay. All rights reserved.

Models & Security

- Terminology
- Access-Control Matrix Model
- Harrison, Ruzzo & Ullman et al.
- Typed Access-Control Model



6

Copyright © 2011 M. E. Kabay. All rights reserved.

Terminology

- Subject: active entity
 - ❑ Subject
 - ❑ User
- Object: passive entity
 - ❑ File
 - ❑ Device
- Right: relation between subject & object
 - ❑ OPEN → subject can establish path for I/O
 - ❑ READ → I/O from file to subject
- Protection State: {rights(subject)}
- Instantiation: specific case realizing model



7

Copyright © 2011 M. E. Kabay. All rights reserved.

Access-Control Matrix Model (1)

| | Process 1 | Process 2 | File 1 | File 2 |
|-----------|-----------|-----------|---------------------------|------------------|
| Process 1 | own | read | read, execute | read, write, own |
| Process 2 | write | own | read, write, execute, own | read |

EXHIBIT 9.1 Example Access-Control Matrix with Two Processes and Two Files

- Model captures protection state CSH5 p 9.4
- Evolution requires *primitive operations*
 - ❑ Primitives are rules for changing matrix
 - ❑ Describe how one adds, changes and removes rights and relations
 - ❑ See next slides

8

Access-Control Matrix Model (2)

➤ Primitives

- **Create subject s** creates a new row and column, both labeled s
- **Create object o** creates a new column labeled o
- **Enter r into $A[s, o]$** adds the right r into the entry in row s and column o ; it corresponds to giving the subject s the right r over the entity o
- **Delete r from $A[s, o]$** removes the right r from the entry in row s and column o ; it corresponds to deleting the subject s 's right r over the entity o
- **Destroy subject s** removes the row and column labeled s
- **Destroy object o** removes the column labeled o

CSH5 p 9.4

Access-Control Matrix Model (3)

➤ Commands

□ Mono-operational: 1 primitive

A *mono-operational* command consists of a single primitive operation. For example, the command

```
command grantwrite(p, f)
  enter write into A[p, f]
end.
```

which gives p write rights over f , is *mono-operational*.

CSH5 p 9.4

□ Conditional

Commands may include conditions. For example, the next command gives the subject p execute rights over a file f if p has read rights over f :

```
command grantexec(p, f)
  if read in A[p, f] then
    enter execute into A[p, f]
  end.
```

If p does not have read rights over f when this command is executed, it does nothing. This command has one condition and so is called *monoconditional*. *Biconditional* commands have two conditions joined by *and*:

```
command copyread(p, q, f)
  if read in A[p, f] and own in A[p, f] then
    enter read into A[q, f]
  end.
```

This command gives a subject q read rights over the object f if the subject p owns f and has read rights over f .

CSH5 p 9.5

Access-Control Matrix Model (4)

➤ Applicability

□ Theoretical basis for 2 widely-used mechanisms:

- ✓ Access-control lists (ACLs)
- ✓ Capability lists

□ For modeling

- ✓ Tool
- ✓ Analyze difficulty of determining level of security

Access-Control Matrix Model (5)

A few more definitions

➤ Command may consist of single primitive operation

```
command grantwrite(p, f)
  enter write into A[p, f]
end.
```

□ Called *Mono-operational* command

➤ Command with only 1 condition: *monoconditional*

```
command grantexec(p, f)
  if read in A[p, f] then
    enter execute into A[p, f]
  end.
```

➤ Command with 2 conditions joined by *and*: *biconditional*

```
command copyread(p, q, f)
  if read in A[p, f] and own in A[p, f] then
    enter read into A[q, f]
  end.
```

➤ System has no commands using *delete* or *destroy** primitives = *monotonic*

* Remember, these operations apply to rules in the matrix, not to data.

Harrison, Ruzzo & Ullman (1)

- How can we test to determine if system is secure?
- Consider
 - ❑ Generic right r for specific entity
 - ❑ Access-control matrix
- So system is secure with respect to r iff
 - ❑ r cannot be added to an entity in matrix
 - ❑ Unless particular subject-object relation includes r as permitted
- Theorems elaborated to establish bounds of provability (see next slide)

iff means
if and only if

Harrison, Ruzzo & Ullman (2)

- **Safety Question:** Is there an algorithm to determine whether a given system with initial state σ is secure with respect to a given right?
- **Theorem (HRU Result):**
 - ❑ Safety question undecidable
 - ❑ Reduce *halting problem* to safety question
 - ✓ Given a description of a program, decide if it will run forever or terminate
 - ✓ Alan Turing proved impossibility of finding general algorithm for this problem
 - ❑ Therefore safety question is also undecidable
- Many other undecidable questions
 - ❑ And a few decidable ones

Typed Access-Control Model

- Adding types to access-control model
 - ❑ Define types of data
 - ❑ Modify rules to allow for finer categories
 - ❑ Model called *TAM*
- Definition: *acyclic rule set*
 - ❑ Entity E or descendents cannot create new entity of same type as E
- Theorem: it is possible to construct an algorithm that will determine if acyclic, monotonic TAMs are secure with respect to generic right r
 - ❑ *But there's no guarantee of finding such an algorithm*



Models & Controls

- Mandatory Access-Control Model
- Discretionary Access-Control Model
- Originator-Controlled Access-Control Model
- Digital Rights Management
- Role-Based Access-Control Models & Groups
- Summary of Models & Controls



Mandatory Access-Control Model



- **MAC sets and enforces access-control rules using a hierarchy of controllers**
 - ❑ US government classification is a MAC
 - ❑ Mandatory because rules must always be followed
 - ❑ Authorities (e.g., ISSO = Information Systems Security Officer) determine rules
 - ❑ System enforces rules
- **Other examples**
 - ❑ MULTICS OS ring-based access-control
 - ❑ Public Key Infrastructure



Mac

17

Copyright © 2011 M. E. Kabay. All rights reserved.

Discretionary Access-Control Model



- **Owners of data have power to determine access controls**
- **DAC**
 - ❑ Most common access control on computers
- **Combinations**
 - ❑ Often see both MAC & DAC
 - ❑ MAC must be applied first
 - ✓ Thus if MAC denies access, no further access or consideration of DAC
 - ✓ But once access granted by MAC, DAC can modify (restrict) further



Unauthorised
access strictly
prohibited

18

Copyright © 2011 M. E. Kabay. All rights reserved.

Originator-Controlled Access-Control Model (1)



- **ORCON**
 - ❑ Originator of data determines access rights
- **Example**
 - ❑ Medical record must be provided to other health-care specialists involved in treatment
 - ❑ But must not be accessible to their staff
- **Formal statement**

More precisely, an originator-controlled access control satisfies two conditions. Suppose an object o is marked as ORCON for organization X . X decides to release o to subjects acting on behalf of another organization Y . Then

1. The subjects to whom the copies of o are given cannot release o to subjects acting on behalf of other organizations without X 's consent; and
2. Any copies of o must bear these restrictions.

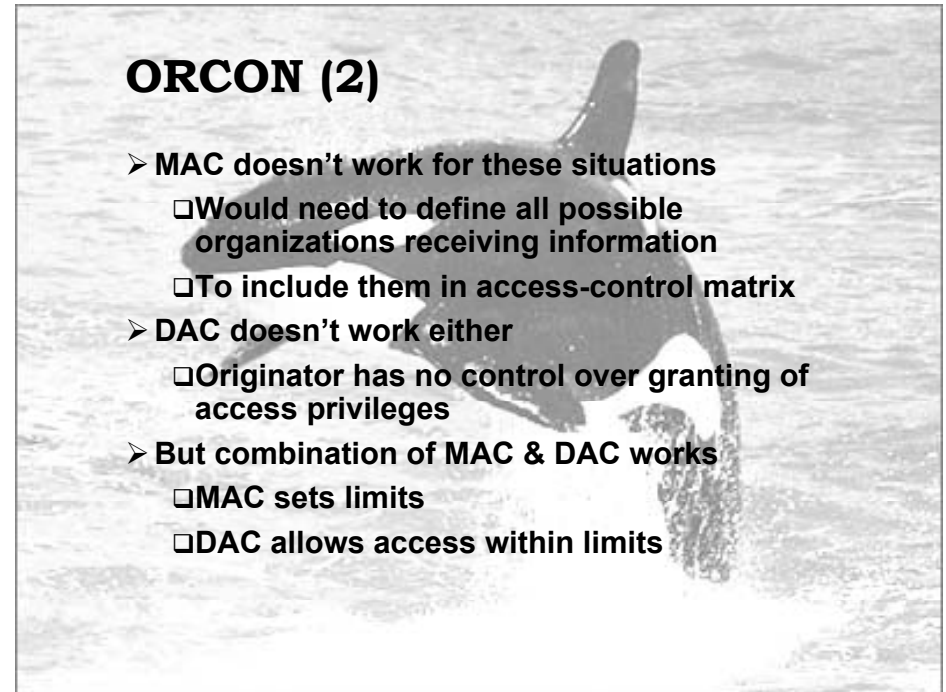
CSH5 p 9.7

19

Copyright © 2011 M. E. Kabay. All rights reserved.

ORCON (2)

- **MAC doesn't work for these situations**
 - ❑ Would need to define all possible organizations receiving information
 - ❑ To include them in access-control matrix
- **DAC doesn't work either**
 - ❑ Originator has no control over granting of access privileges
- **But combination of MAC & DAC works**
 - ❑ MAC sets limits
 - ❑ DAC allows access within limits



ORCON & DRM

- Digital rights management illustrates ORCON
- Copyright owners want to control distribution after they sell license to use materials
 - ❑ Music, video, software....
- Fundamental problem
 - ❑ Access controls apply to *entities* (files, devices, music...)
 - ❑ ORCON applies to *information*
 - ❑ *Once information has been accessed, it is not possible to control distribution through alternate channels*
 - ✓ E.g., CSH5 electronic copy is protected against data extraction...
 - ✓ ...but not against screen shots, as you can see in this presentation.... ☺



captain orcon
Taking broadband to the next level

21

Copyright © 2011 M. E. Kabay. All rights reserved.

Role-Based Access-Control (RBAC) Models & Groups

- Situations often require access controls defined for functional groups
 - ❑ Bookkeepers, doctors, nurses, engineers, researchers, administrators....
 - ❑ So define
 - ✓ *Authorized roles* for each subject
 - ✓ *One active role* per subject at any 1 time
- Example: separation of duties
 - ❑ Multiple roles must combine efforts to perform task signing check for >\$50K
 - ❑ RBAC defines rule preventing same subject from having same role at same time



22

Copyright © 2011 M. E. Kabay. All rights reserved.

Summary of Models & Controls

- Fundamental difference in orientation
 - ❑ MAC, DAC, & ORCON are data-centric
 - ❑ RBAC is focused on subject's needs
- Principle of least privilege
 - ❑ Assign minimum set of privileges for successful work
 - ❑ RBAC constrains set of commands for each subject
 - ❑ MAC, DAC, & ORCON set attributes on data
- Several models have been created that apply combinations of these approaches (see following)

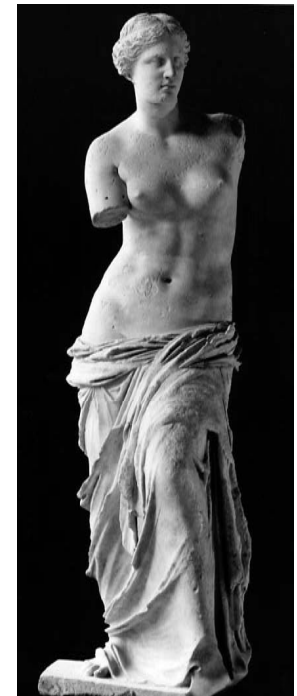


23

Copyright © 2011 M. E. Kabay. All rights reserved.

Classic Models

- Bell-LaPadula Model
- Biba's Strict Integrity Policy Model
- Clark-Wilson Model
- Chinese Wall Model
- Summary of Classic Models



24

Copyright © 2011 M. E. Kabay. All rights reserved.

Bell-LaPadula Model (1)

➤ Formalization of government system

- ❑ Looks at *confidentiality*
- ❑ UNCLASSIFIED, CONFIDENTIAL, SECRET, & TOP SECRET levels for information



- ❑ Example of *multilevel security model*

➤ Terminology

- ❑ UNCLASSIFIED = lowest security level
- ❑ Subject *cleared* into level → *security clearance = level(s)*
- ❑ Object *classified* at level → *security classification = level(o)*
- ❑ Goal: prevent leakage (information flow from high security classification to lower)

25

Copyright © 2011 M. E. Kabay. All rights reserved.

Bell-LaPadula Model (2)

➤ Example

- ❑ Documents are classified
 - ✓ *Paper on Norwich U* is CONFIDENTIAL
 - ✓ *Article on Ecoterrorists* is SECRET
 - ✓ *Book on Al Qaeda* is TOP SECRET
- ❑ Tom cleared to SECRET level

➤ Then

- ❑ Tom can read WHAT?
- ❑ And Tom cannot read WHAT?



26

Copyright © 2011 M. E. Kabay. All rights reserved.

Bell-LaPadula Model (3)

➤ Simple security property

- ❑ Subject *s* can read object *o* iff $level(o) \leq level(s)$
- ❑ AKA *no-reads-up* rule
- ❑ Cannot stop leakage if higher-clearance writes to lower-classification medium

➤ *-property (pronounced *star property*)

- ❑ *s* can write *o* iff $level(s) \leq level(o)$

- ❑ AKA *no-writes-down* rule

➤ Discretionary Security Property

- ❑ *s* can read *o* iff access-control matrix entry for *s* & *o* contains READ right



27

Copyright © 2011 M. E. Kabay. All rights reserved.

Bell-LaPadula Model (4)

➤ Basic Security Theorem

- ❑ If a system starts in a secure state
- ❑ And if every command obeys all of these:
 - ✓ Simple security property
 - ✓ *-property
 - ✓ Discretionary security property
- ❑ Then the system will remain secure



Basic Security Theorem. Let a system Σ have a secure initial state σ_0 . Further, let every command in this system obey the simple security property, the *-property, and the discretionary security property. Then every state $\sigma_i, i \geq 0$, is also secure.

CSH5 p 9.10

28

Copyright © 2011 M. E. Kabay. All rights reserved.

Bell-LaPadula Model (5)



- Compartments
 - ❑ Category = expansion of security level
 - ❑ Security compartment = (level, category set)
 - ✓ E.g., $level(EurDoc) = (CONFIDENTIAL, \{EUR\})$
 - ✓ $level(EurAsiaDoc) = (SECRET, \{EUR, ASIA\})$
 - ❑ Cannot talk about $>$ or $<$ because categories no longer linearly ordered
- Define relation *dominates* = *dom*

We compare compartments using the relation *dom*, for “dominates.”
Definition. Let L and L' be security levels and let C and C' be category sets. Then $(L, C) dom (L', C')$ if and only if $L' \leq L$ and $C' \subseteq C$

The *dom* relation plays the role that “greater than or equal to” did for security levels. Continuing our example, $level(Erin) = (SECRET, \{EUR\}) dom (CONFIDENTIAL, \{EUR\}) = level(EurDoc)$, and $level(EurAsiaDoc) = (SECRET, \{EUR, ASIA\}) dom (SECRET, \{EUR\}) = level(Erin)$.

We now reformulate the simple security property and *-property in terms of *dom*:
Definition. The *simple security property* says that a subject s can read an object o if and only if $level(s) dom level(o)$.
Definition. The **-property* says that a subject s can write to an object o if and only if $level(o) dom level(s)$.

29

Bell-LaPadula Model (6)



- Determining highest security compartment that two subjects can read and lowest they can both write
- Properties of compartments
 - ❑ Reflexive property: $level(s) dom level(s)$
 - ❑ Antisymmetric property:
 - ✓ If both $level(s) dom level(o)$ & $level(o) dom level(s)$ are true, then $level(s) = level(o)$
 - ❑ Transitive property:
 - ✓ If $level(s1) dom level(o)$ & $level(o) dom level(s2)$, then $level(s1) dom level(s2)$



30

Copyright © 2011 M. E. Kabay. All rights reserved.

Bell-LaPadula Model (7)

Turning it into plain English

- Reflexive: If accounting files are confidential
 - ❑ Then any accounting file is confidential
- Antisymmetric: If Alice can access the accounting files
 - ❑ And accounting files are at the security level of Alice,
 - ❑ Then Alice's security level and the accounting files' security level are equivalent
- Transitive:
 - ❑ If Alice can access the accounting files
 - ❑ And the accounting files are of higher security than the personnel files
 - ❑ Then Alice can access the personnel files



31

Copyright © 2011 M. E. Kabay. All rights reserved.

Bell-LaPadula Model (8)



Prof Matt Bishop writes:

- The influence of the Bell-LaPadula Model permeates all policy modeling in computer security.
- It was the first mathematical model to capture attributes of a real system in its rules.
- It formed the basis for several standards, including the [DoD's] *Trusted Computer System Evaluation Criteria* (the TCSEC or the “Orange Book”....)
- Even in controversy, the model spurred further studies in the foundations of computer security.



Bishop, M. (2003). *Computer Security: Art and Science*. Addison-Wesley (ISBN 0-201-44099-7). xii + 1084. Index. P 148.

32

Copyright © 2011 M. E. Kabay. All rights reserved.

Biba's Strict Integrity Policy Model (1)



- Instead of *confidentiality*, consider *trustworthiness*
- Define *integrity classes* $i\text{-level}(s)$
- Simple integrity property:
 - ❑ Subject s can read object o iff $i\text{-level}(o) \text{ dom } i\text{-level}(s)$
 - ❑ Allows reads up and disallows reads down
- *-integrity property
 - ❑ Subject s can write to object o iff $i\text{-level}(s) \text{ dom } i\text{-level}(o)$
 - ❑ Allows writes down and disallows writes up

WHY?
Explain.

WHY?
Explain.

Biba's Model (2)

- Execution integrity property
 - ❑ Subject s can execute subject s' iff $i\text{-level}(s') \text{ dom } i\text{-level}(s)$
 - ❑ So process can pass data only to less trustworthy (or equally trustworthy) process
 - ❑ Cannot pass data to more trustworthy process

WHY?
Explain.

Given these three properties, one can show:
Theorem. If information can be transferred from object o_1 to object o_n , then by the simple integrity property, the *-integrity property, and the execution integrity property, $i\text{-level}(o_1) \text{ dom } i\text{-level}(o_n)$.
 In other words, if all the rules of Biba's model are followed, the integrity of information cannot be corrupted because information can never flow from a less trustworthy object to a more trustworthy object.

CSH5 p 9.13

Biba's Model (3)

- Suggests method for testing programs
- Label all data with tags indicating level of trust
- Logic in program keeps track of trustworthiness
 - ❑ Changes label according to Biba's model
- Violations of rules → exceptions
 - ❑ Abort
 - ❑ Log warning
 - ❑ Error message
- Useful in improving security of programming

Clark-Wilson Model (1)

- Lipner's Rules for Commercial Integrity Models

1. Users may not write their own programs to manipulate trusted data. Instead, they must use programs authorized to access that data.
2. Programmers develop and test programs on nonproduction systems, using non-production copies of production data if necessary.
3. Moving a program from nonproduction systems to production systems requires a special process.
4. That special process must be controlled and audited.
5. Managers and system auditors must have access to system logs and the system's current state.

CSH5 p 9.14

Clark-Wilson Model (2)

Concepts and terminology

- Integrity constraints: rules about data relations (e.g., grades in gradebook must match grades on tests)
- Consistent state: all integrity constraints met
- Well-formed transactions (WFTs): moving from one consistent state to another (e.g., filling in all the grades from a test, not just some)
- Integrity verification procedures (IVPs): methods for checking that integrity constraints are met
- Constrained data items (CDIs): data subject to integrity constraints (all others are unconstrained data items – UDIs)
- Transformation procedures (TPs): Functions defining WFTs must be *certified* to be well formed and correctly implemented

Clark-Wilson Model (3)

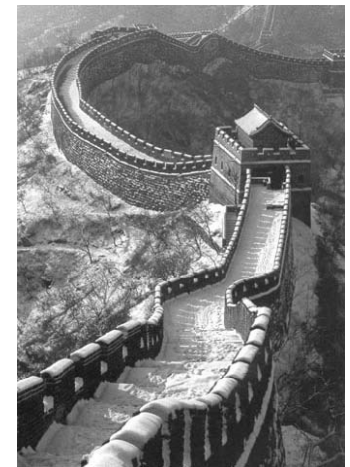
- Nine rules
 - ❑5 for certification of data & TPs
 - ❑4 for enforcement of certifications
- Certification Rule 1: IVP ensures consistent state at all times
- Certification Rule 2: TP maintain valid states
- Enforcement Rule 1: Only certified TPs may manipulate associated CDIs
- Enforcement Rule 2: Every user must be associated with specific TPs and CDIs to be granted access – implies identification & authentication

Clark-Wilson Model (4)

- Enforcement Rule 3: Every user must be authenticated before executing a TP
- Certification Rule 3: Separation of duty is essential
- Certification Rules 4: Logging is essential and must allow reconstruction of transactions
- Certification Rule 5: Any TP working with UDI must either reject the UDI or transform it into a CDI
- Enforcement Rule 4: As part of the separation of duties principle, only the certifier of a TP may modify its scope; and no certifier may execute the TP.
- The Clark-Wilson model reflects real-world security policies
 - ❑But a certifier may fraudulently claim to have applied proper certification procedures

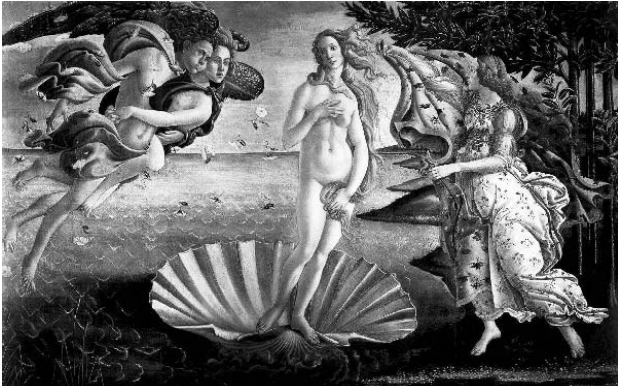
Chinese Wall Model

- AKA Brewer-Nash model
- Goal: prevent conflicts of interest
- How?
 - ❑Group data into *company data sets*
 - ❑Group company data sets into *conflict-of-interest classes*
- Rule: if entities are in same conflict-of-interest class, subject cannot read both classes
 - ❑Typical application: attorneys
- Sanitized class has all confidential content removed



Summary of Classic Models

- *Bell LaPadula* model describes widely-used scheme for protecting confidentiality
- *Biba* model focuses on trustworthiness and trust
- *Clark-Wilson* model incorporates process integrity
- *Chinese Wall* model includes conflicts of interest



*The Birth of Venus
(Nascita di Venere)*
Sandro Botticelli
c. 1486
Uffizi Gallery,
Florence, Italy

41

Copyright © 2011 M. E. Kabay. All rights reserved.

Other Models

Specific models applied in specific contexts

- Clinical Information Systems Security Model
- Traducement model for real estate
- Noninterference security – prevent LOW subject from acquiring HIGH data at particular time – but allow HIGH subject to choose to send HIGH data later
- Deducibility security – prevent inference by LOW subjects

42

Copyright © 2011 M. E. Kabay. All rights reserved.

Conclusions

- Usefulness of the models depends on thoroughness of systems analysis
 - ❑ Missing details may invalidate application of model
- Crucial role in advanced theoretical research in today's information security
- Especially important for mathematical proof of security in specific well-defined systems



43

Copyright © 2011 M. E. Kabay. All rights reserved.

DISCUSSION

44

Copyright © 2011 M. E. Kabay. All rights reserved.