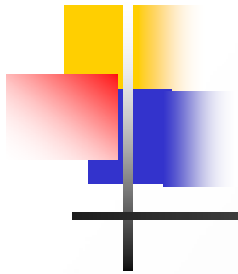
Trust Model
Supplements and Taxonomy

Vasily Desnitsky and Igor Kotenko

Computer Security Research Group,
St. Petersburg Institute for Informatics and
Automation of Russian Academy of Sciences

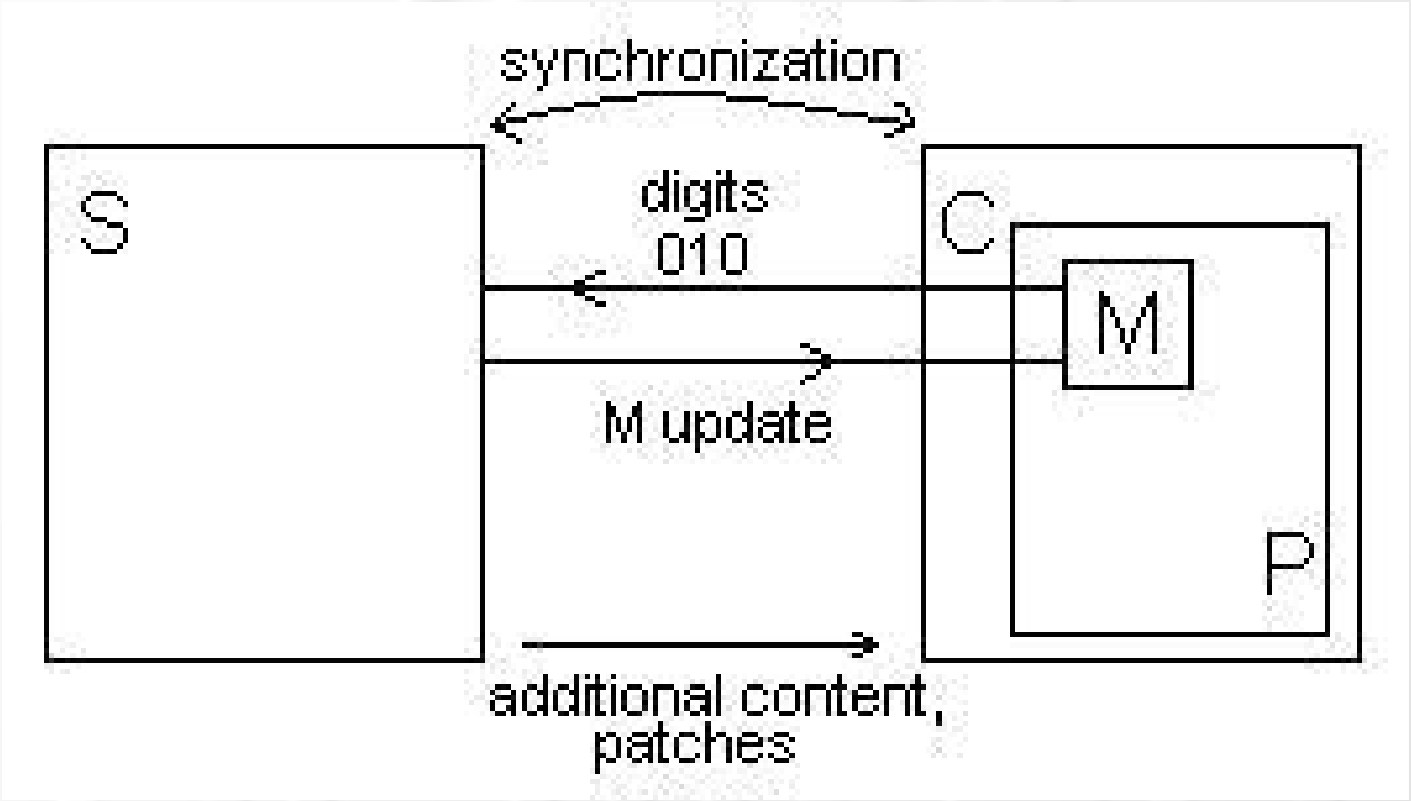


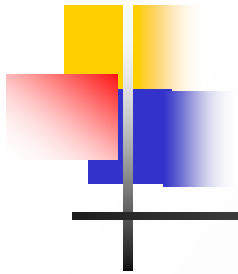
Contents

1. Introduction
2. Distribution of verification functions between client program and trusted server
3. Monitor update
4. Actions executed by trusted server and/or by client when detecting a client program tampering
5. Client program life cycle
6. Trust model taxonomy
7. Conclusion

Introduction

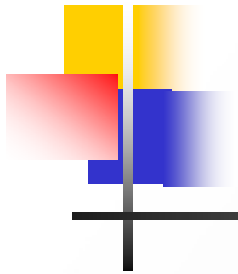
- Trust Model





Distribution of verification functions between client program and trusted server (1/2)

- It is possible to execute some verification functions within the trusted server
- Client monitor has two responsibilities:
  - To gather needed data about a program and its state
  - To execute a part of verification functions
- Trusted server has two responsibilities:
  - To interpret all received signatures to authenticate the client program
  - To get and extract gathered data from client to execute some verification functions itself



Distribution of verification functions between client program and trusted server (2/2)

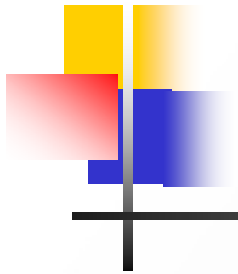
- Problem: what verifications should be executed on
  - trusted server?
  - client monitor?
  - both?
- To approach the problem solution, we may take into account the following verification properties:
  - computational complexity
  - relative importance
  - size of input data and their structure

Monitor update (1/2)

- We need to construct an infinite sequence of monitors *M*0, *M*1, *M*2, ... so that
  - **{$\exists$ const1 > 0: $\forall$ k $\geq$ 0 time_of_breaking(*M*k) > const1 }**
    – monitor level

    Or
  - **$\forall$ i $\geq$ 0**

    {$\exists$ const2 > 0: $\forall$ k $\geq$ 0 time_of_breaking(VFi(Mk)) > const2}
    – verification function level; where *VFi(Mk)* is some verification number *i* of *Mk* monitor
- So our aim is to design appropriate sequences of verification functions and also data gather functions



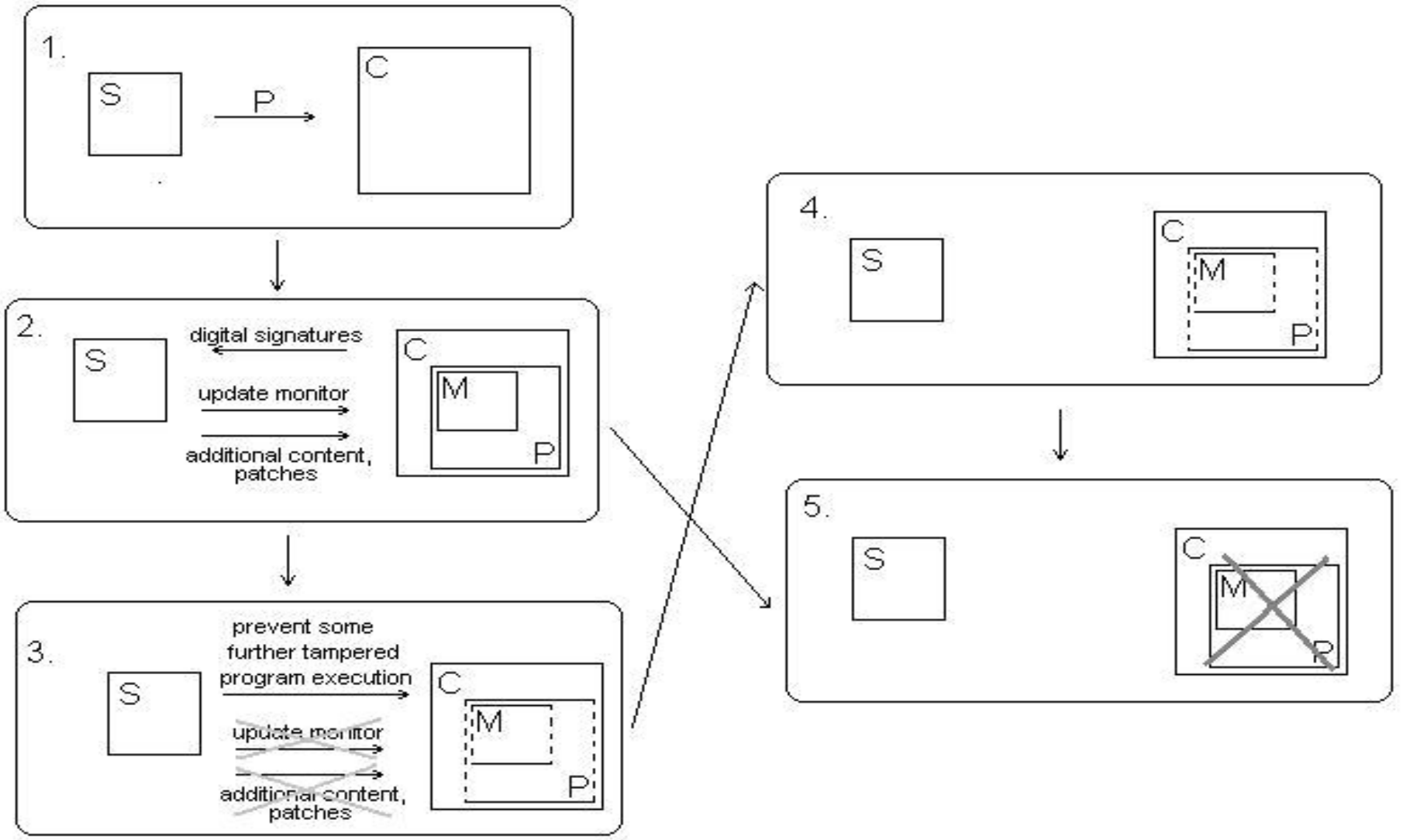
Monitor update (2/2)

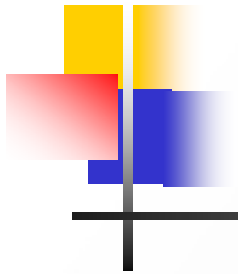
- Each verification function may defer from its previous versions by some
  - **regular modifications**, e.g.
    - modification of secret key
    - hash function modification
    - sequence and amount of gathered data
    - sequence of passing Control Flow Graph
  - and **qualitative modifications**, such as
    - adding a new type of verifications and gathering functions
    - deleting some existing out-of-date verifications and gathering functions

Actions executed by trusted server and/or by client when detecting a client program tampering

- Suspend to send additional content, patches and system data
- Suspend the monitor update
- Send to the client certain commands that execute some irreversible program modifications preventing the program work:
  - destruction of the program parts which are difficult to restore;
  - definite or arbitrary modification of program;
  - definite or arbitrary modification or removal of data and files used by program.

Client program life cycle





Trust model taxonomy (1/4)

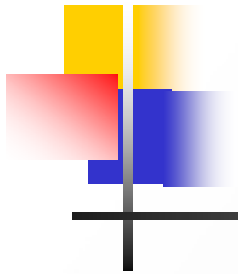
- In the current version of taxonomy there are three high-level notions:
  - Client
  - Trusted server
  - Data streams between them

Trust model taxonomy (2/4)

- Client
  - Target program
    - Monitor
      - Verification functions
        - Static verifications
        - Integrity verification of program code; implementation: check-sums and hash-functions
        - Dynamic verification
        - Calculation results correctness
        - Assert-instructions
        - Environment verification: HW/OS
        - CFG correctness verification. Notions: basic blocks, marks on them, CFG paths
      - Functions of data gathering for verification on server
      - Reaction on illegal modifications
        - Destruction of difficult restoring parts of program
        - Definite or arbitrary program modification
        - Definite or arbitrary program modification or removal of data and files used by program
    - Proper client program

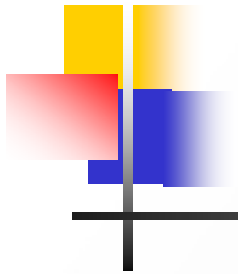
Trust model taxonomy (3/4)

- Server
  - Signature interpreting component
  - Monitor factory: mechanism, routinely generating a new monitor version
    - Regular modification
      - Secret key modification
      - Hash function modification
      - Modification of sequence and amount of gathered data
      - Modification of sequence of passing vertex of Control Flow Graph
    - Qualitative modification
      - Addition of new verifications type
      - Addition of new gathering functions type
      - Removal out-of-date verifications and gathering functions (for time optimization)
  - Manager, determining “intension” of entire protection functioning:
    - monitor update periodicity
    - periodicity of receiving required signatures
    - working life of signatures and others
    - current state of client (tampered/not tampered)
  - Verification functions



Trust model taxonomy (4/4)

- Data streams
  - Streams from client to server
    - Authentication signatures
    - Data for verification on server
  - Streams from server to client
    - Monitor updates
    - Additional content and patches
  - System data, organizing a single physical data stream between client and server and synchronizing client with server



Conclusion

- So, we have touched several unsolved problems:
  - How to distribute verification functions between client program and trusted server?
  - What methods may be used to construct verification sequences for monitor update?
  - The problem on acceptability of additional preventing actions when detecting a client program tampering.