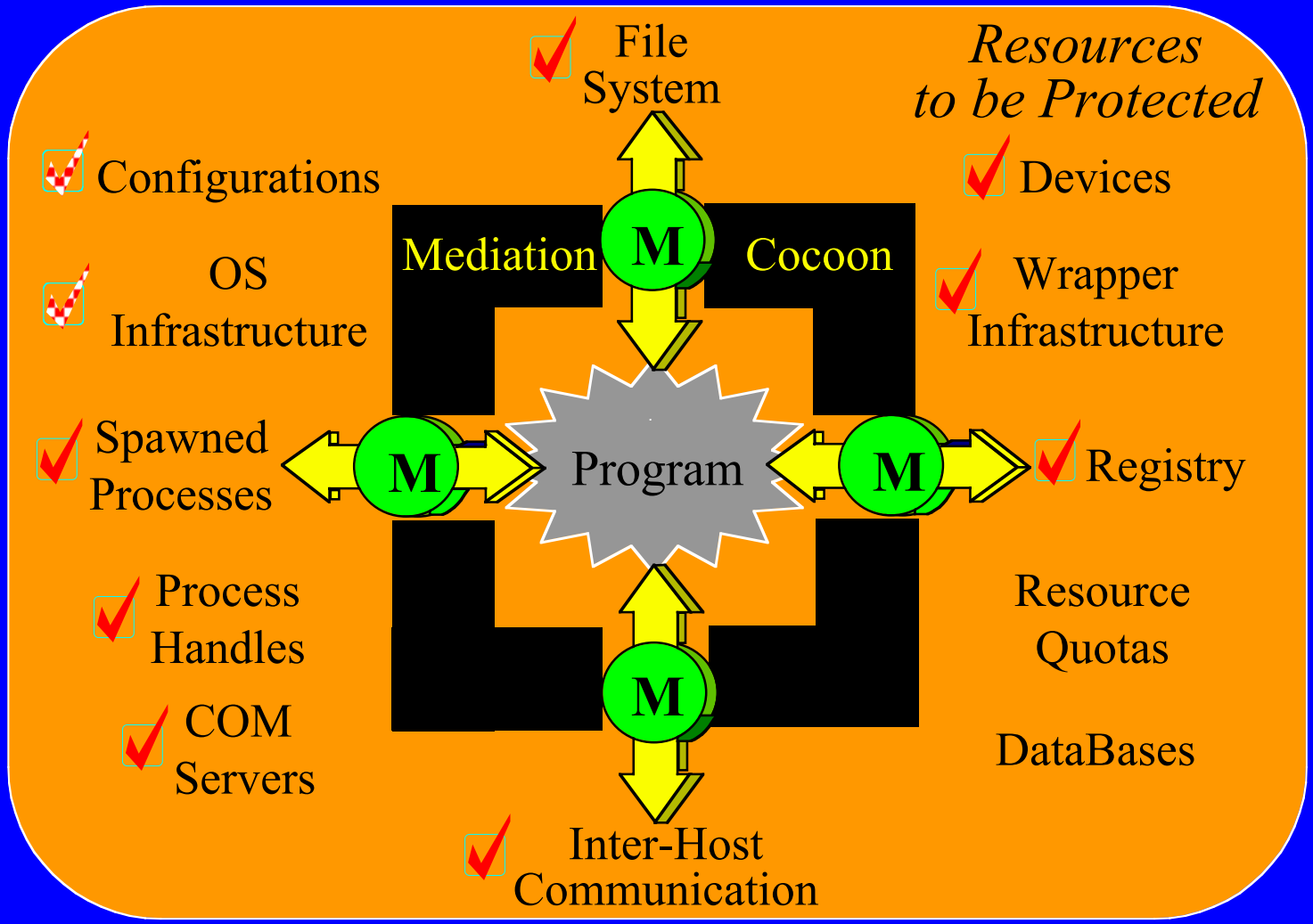
Possibly Safely Executing Malicious Code Within COTS Products

WG10.4 Workshop June 28, 2002

Bob Balzer
Teknowledge
balzer@teknowledge.com

TEKNOWLEDGE

Wrapper Defenses

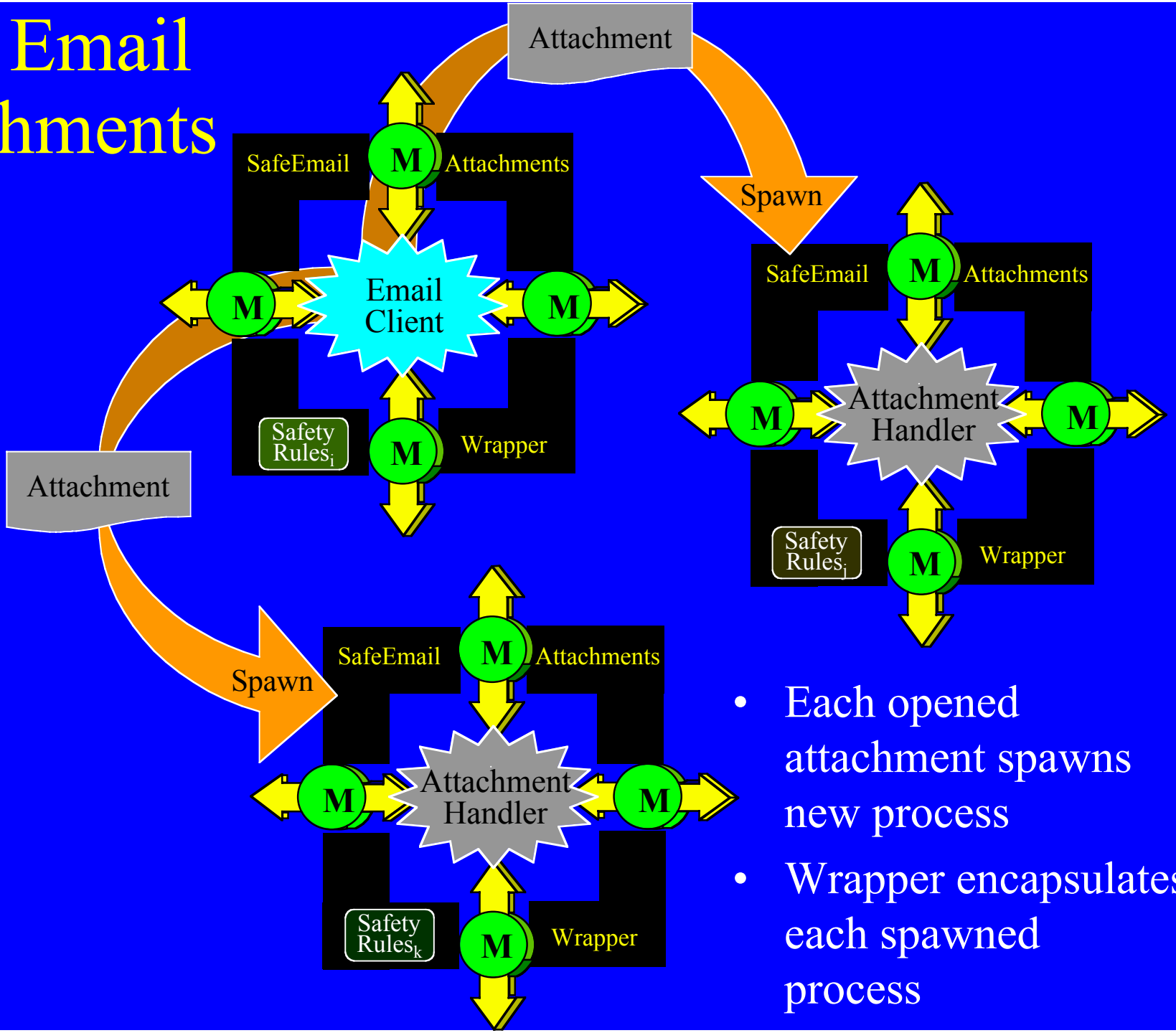


- Separate *possibly malicious* program(s) from resources
  - Mediate their interactions with those resources

  ⇒ Wrap every *possibly malicious* program

  Wrap process initiators and propagate wrappers into spawned descendants

Safe Email Attachments

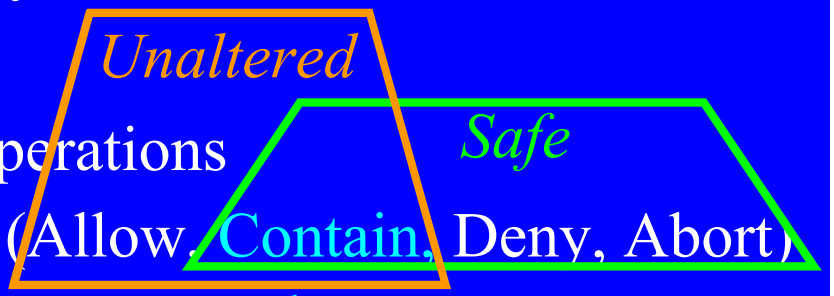


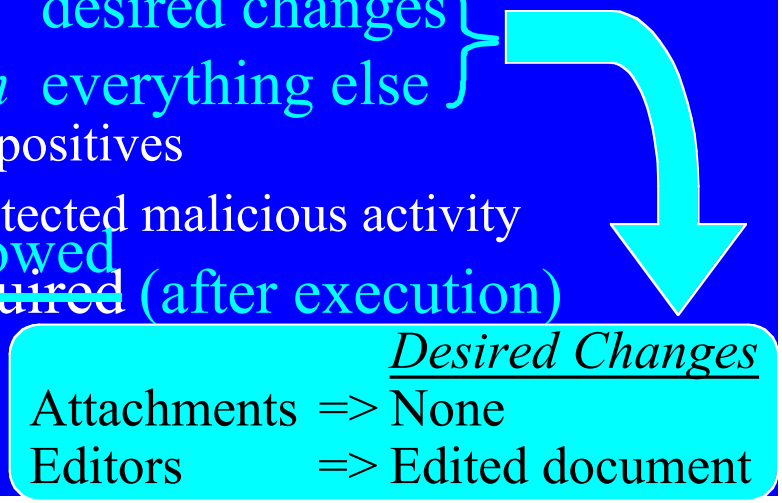
- Each opened attachment spawns new process
- Wrapper encapsulates each spawned process

Safe-Email Attachments Demo

Contained ~~Wrapped~~ Execution

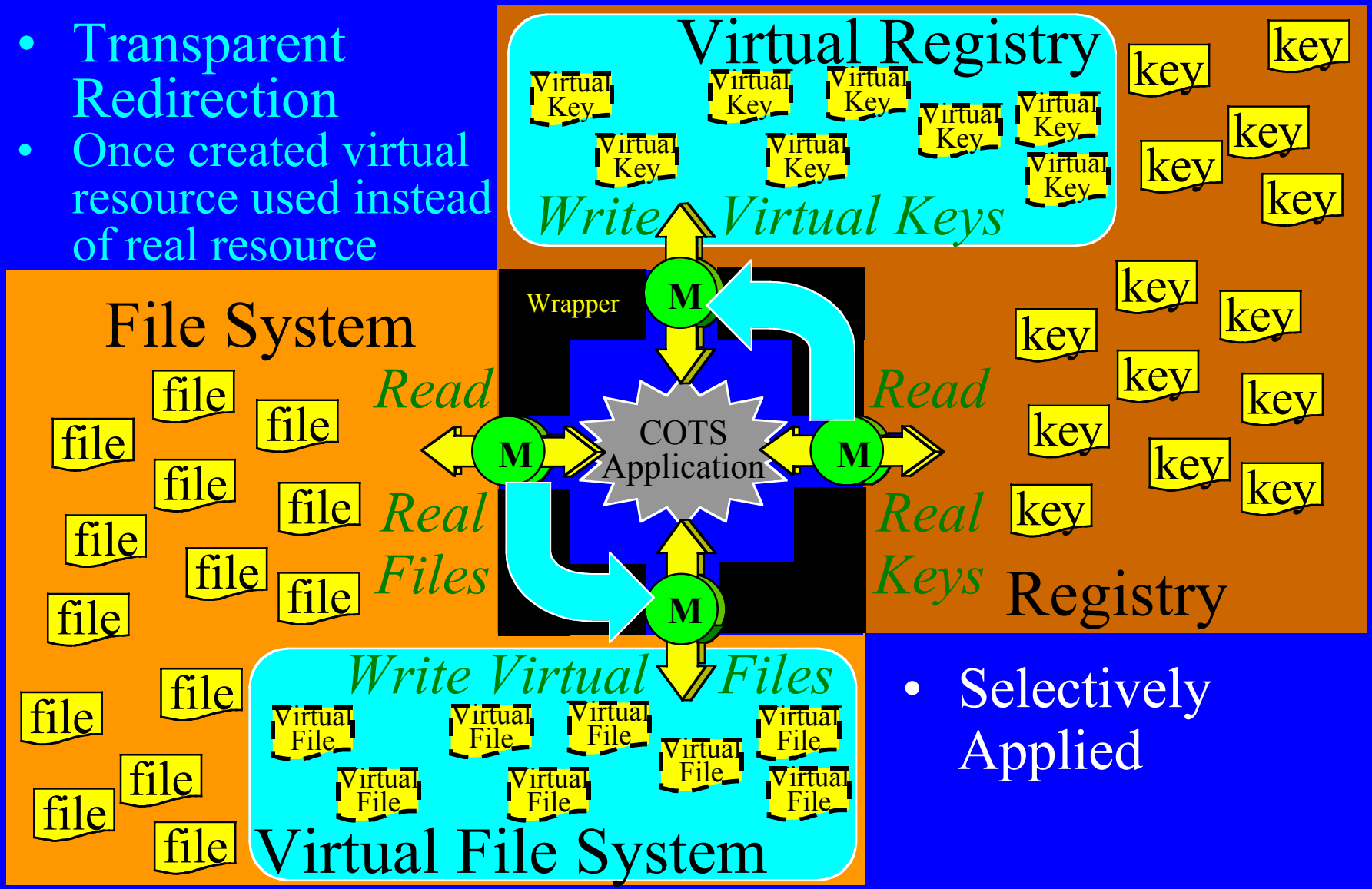
- Goal: Safely Execute possibly Malicious Code
- Approach:
  - Mediate potentially harmful operations
  - Apply Authorization function (Allow, Contain, Deny, Abort)
  - Contained operations only affect wrapped process
- Problems
  - Configuration ~~difficult~~ easy { *Allow* desired changes; *Contain* everything else }
    - Tight policy generates many false positives
    - Loose policy leaves room for undetected malicious activity
  - ~~Early~~ Late authorization decision ~~required~~ allowed (after execution)





Contained Execution

- Transparent Redirection
- Once created virtual resource used instead of real resource



- Selectively Applied

Contained Execution Demo

Contained Execution

- Like a Virtual Machine
  - Execution is isolated
- Unlike a Virtual Machine
  - Process-Level (instead of machine-level)
  - Selective (instead of copying entire environment)
  - Incremental (copies created as needed)

Contained Execution
(contain selected modifications within process)

- Contained Resource (currently implemented)
  - Virtual Registry (selected changes made to virtual keys)
  - Virtual File System (selected changes made to virtual files)
- Benefits:
  - Program Execution has no effect on rest of system
    - ⇒Blocks single-stage attacks (no effect on rest of system)
    - ⇒Blocks multi -stage attacks (no transfer of aggregated effects)
  - Rule violations can be safely contained and auto-authorized
  - Attack determination can be safely delayed
    - More behavior analyzed => better decision
    - Supports autonomic responses
    - Reduced false alerts
  - Can rerun information extraction attacks with misinformation

Hardened Defenses

- NonBypassability
  - Alternative paths to OS service (other user APIs)
  - Lower-level APIs
    - NTDLL (normal, but undocumented, API)
    - Hardware Call-Gate
- Secure (Interactive) Alerts
  - Windows are not securable objects
- Self-Protection (in same address space)
  - Rules to protect persistent data (files)
  - Memory protect for loaded data
  - (Eventually) In-Line Reference Monitor [Schneider]