9th International Common Criteria Conference – Designing the Trusted Service Bus for EAL5

David Ochel, atsec information security

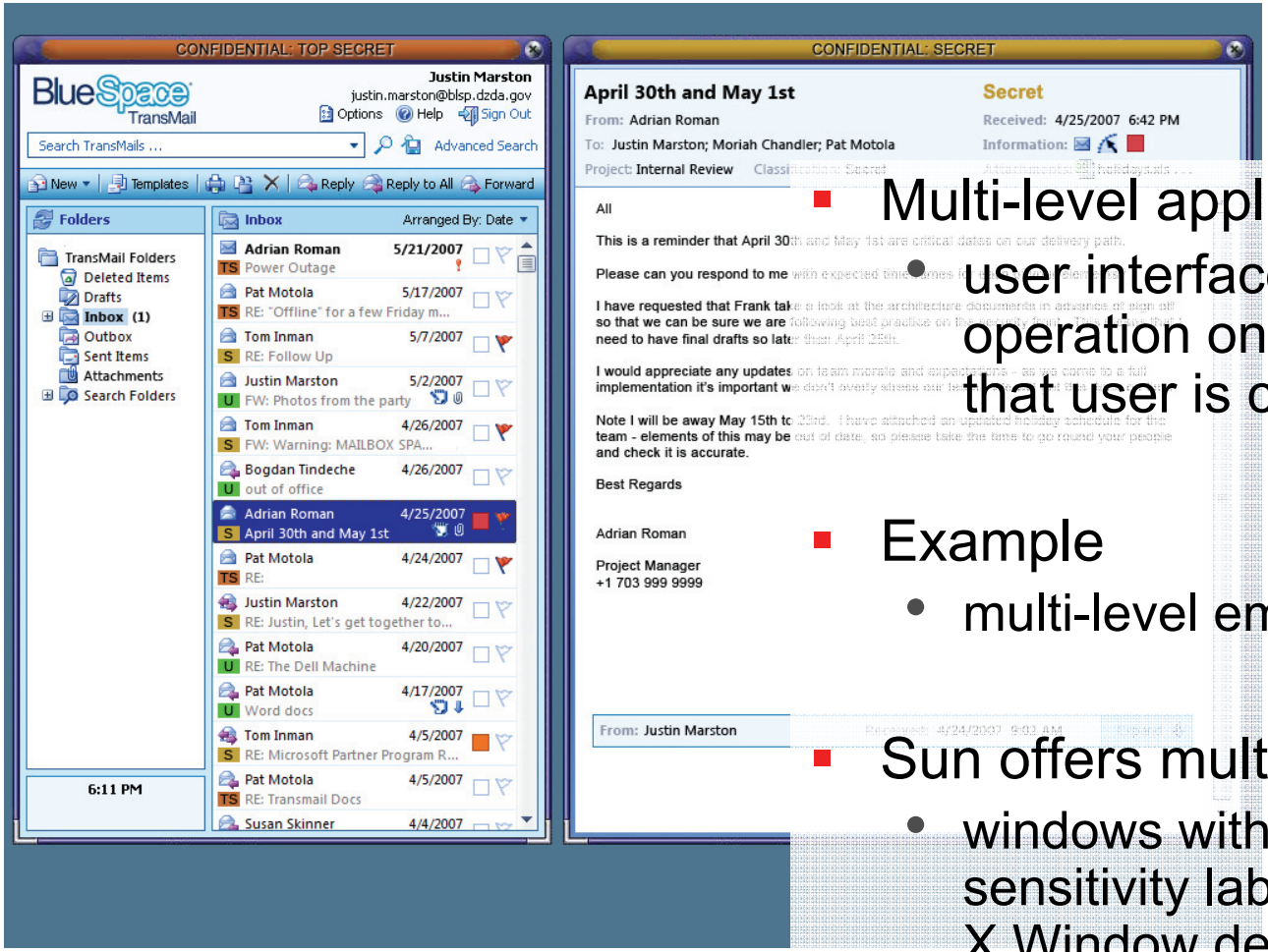
Brian Vetter, BlueSpace Software

atsec the information security provider

Agenda

- Objective
  - Development of multi-level applications
- Background
  - Solaris Trusted Extensions: zone concept
  - Zones and multi-level applications
- Trusted Service Bus
  - Objectives
  - Architecture and design
  - Results and implementation status

Objective: Facilitate multi-level applications on Solaris Trusted Extensions



- Multi-level applications
  - user interface integrates operation on multiple levels that user is cleared for
- Example
  - multi-level email client
- Sun offers multi-level desktop
  - windows with different sensitivity labels on the same X Window desktop

Background: Sun Solaris Trusted Extensions

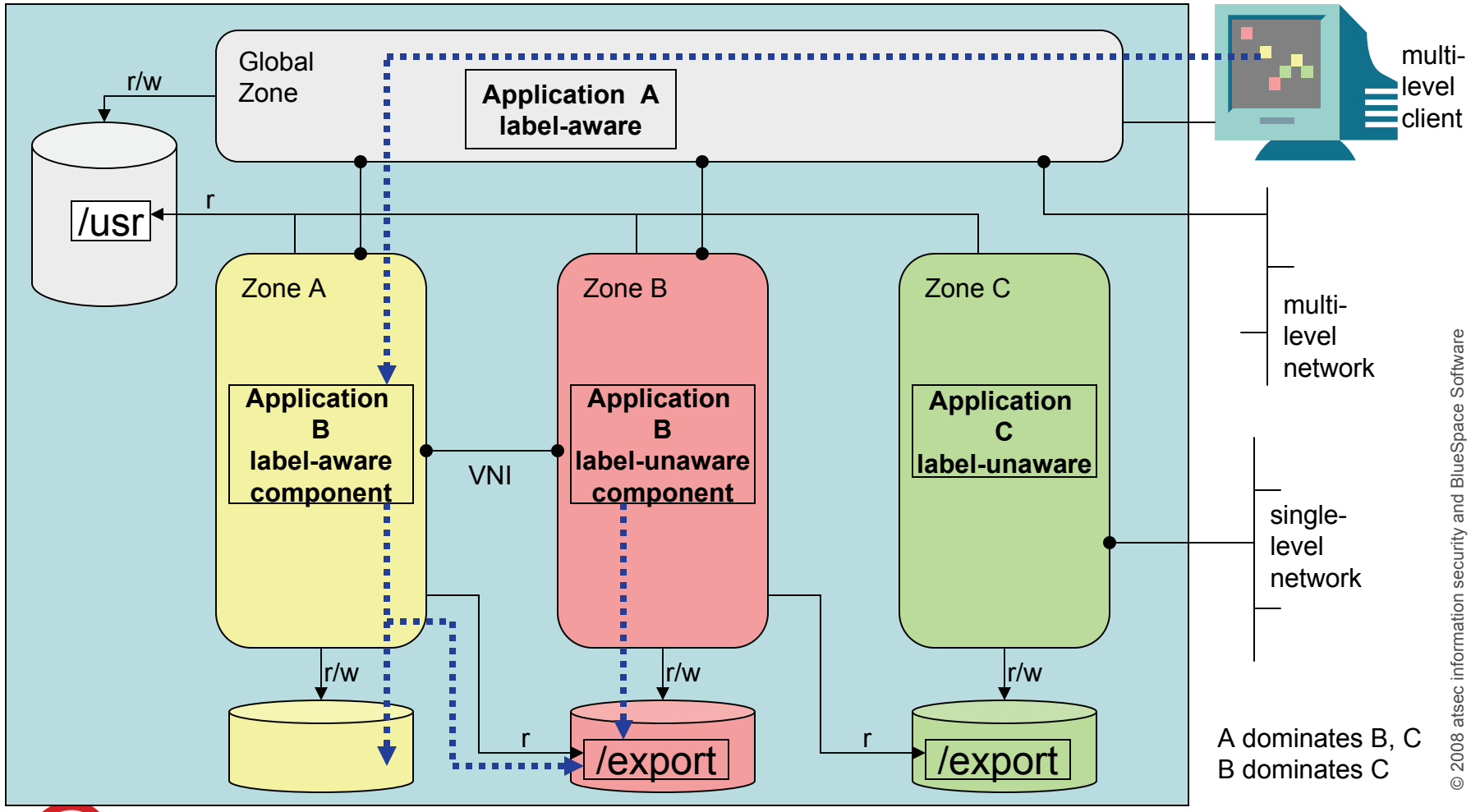
- certified June 11, 2008 at EAL4 (CCS – Canada)

- implements “zone” concept
  - one virtualized runtime environment per sensitivity label
    - one zone per label – labeling handled by OS
    - file system and network resources associated with zone
    - no need for processes and resources to be “label-aware”
    - communication between zones subject to MAC enforcement
  - one “global zone”
    - central management of TSF and labeled zones
    - administrative multi-level environment
    - exports system and other files as read-only to other zones

Background: Communication between zones

- MAC enforcement between zones (LSPP)
  - read-down possible via file system (loopback mount in dominating zone)
  - write-up possible via named pipes
  - read-up not possible
  - write-down not possible
- exception: multi-level network ports
  - exempt from MAC enforcement
  - sending and receiving process both require net_bindmlp privilege

Background: Zone concept



Background: Zones and multi-level applications

- problem
  - no write-down between application instances running in different zones, but needed for multi-level applications
- obvious solution
  - develop a label-aware application running in global zone, or
  - give application instances in labeled zones net_bindmlp privilege
- problem with the obvious solutions
  - OS-enforced mandatory access control is circumvented
    - it is up to applications to limit information flow
  - application may be large and/or complex
    - huge attack surface
    - difficult to evaluate
    - system accreditation difficult

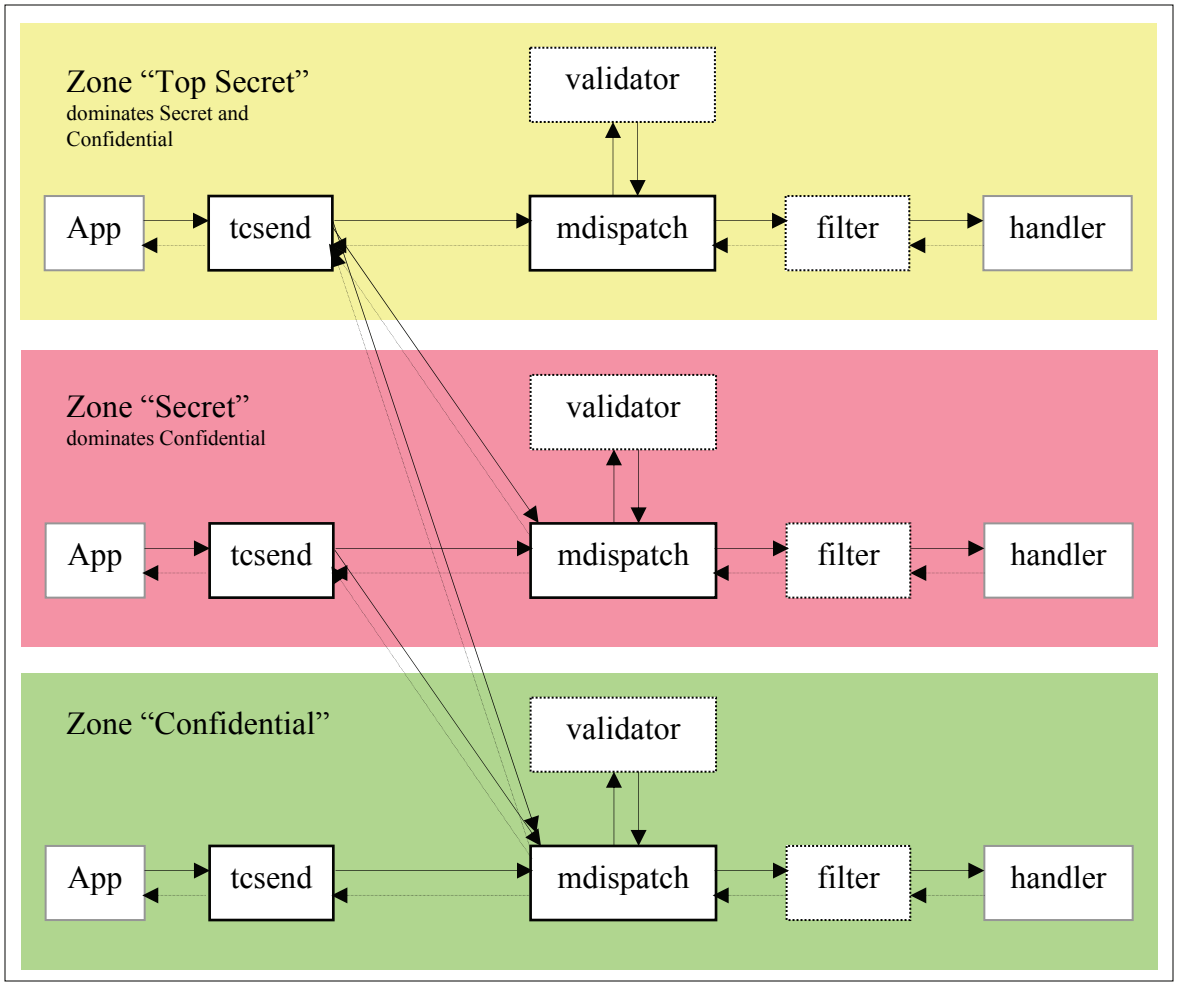
Trusted Service Bus: Objectives

- multi-level email client/server solution
- original architecture:
  - majority of code in global zone
  - processing email with different sensitivity labels
  - serving multi-level clients
  - application-enforced mandatory access
- objectives:
  - reduce complexity and attack surface
  - reduce certification and accreditation footprint

Trusted Service Bus: Architecture and design I

- run application instances in labeled zones
  - application does not need to be label-aware
  - communication via multi-level port
- separate privileged code from rest of application
  - small component with net_bindmlp privilege
- limit information exchange via net_bindmlp port
  - only specific message formats
  - ability to further filter and validate message contents (application- and consumer-defined validators/filters)
  - write-down only (applications use OS functions for read-down)

Trusted Service Bus: Architecture and design II



- Trusted Service Bus
  - uses net_bindmlp privilege
  - sender process (tcsend)
  - receiver process (mdispatch)
- application
  - uses Trusted Service Bus
  - provides validator scripts for compliance to message format
  - invokes handler (app instance in receiving zone)
- consumers
  - can "plug in" additional filters

Trusted Service Bus: Architecture and design III

- Use of and dependencies on OS-provided functionality
  - virtual network interfaces
    - communication not exposed to physical network
  - peer credentials for communicating processes
    - real and effective user/group ID, label, privileges
  - auditing
    - audit records generated with OS-provided functionality
  - read-down mounts of file systems
    - Trusted Service Bus does not need to implement this
  - protection mechanisms
    - process separation, privilege enforcement, zones, …

Trusted Service Bus: Objectives met?

- reduced attack surface and evaluation effort:
  - small amount of code
    - easy to comprehend, document, and evaluate
    - straightforward functionality and controls that can be understood and accredited by consumers
  - where possible, use already certified OS functionality
  - component is separate from non-security relevant code

- side effects:
  - re-usable, application-independent component

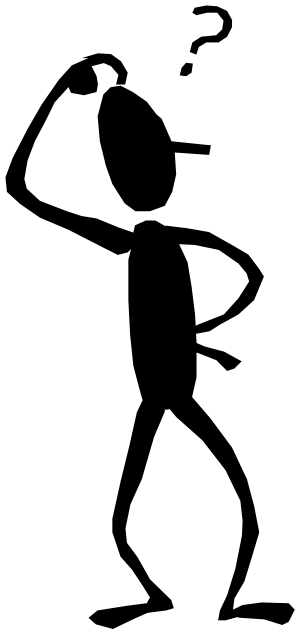
Trusted Service Bus: Implementation status

- Implemented
- Integrated into an application
- Security Target complete
- Application for certification expected soon

References

- atsec; BlueSpace: Trusted Service Bus Security Target. Version 0.93, 2008-07-29.
- Faden, G.: Multilevel Filesystems in Solaris Trusted Extensions. 2007.
  - http://opensolaris.org/os/community/security/projects/tx/sacmat04s-faden-1.pdf
- Faden, G.: Solaris Trusted Extensions, Architectural Overview. April 2006.
  - http://www.opensolaris.org/os/community/security/projects/tx/TrustedExtensionsArch.pdf
- NSA Information Systems Security Organization: Labeled Security Protection Profile. Version 1.b, 8 October 1999.
  - http://www.niap-ccevs.org/cc-scheme/pp/id/pp_os_ls_v1.b
- Sun Microsystems: Solaris 10 11/06 Trusted Extensions Security Target. Version 1.2, 30 April, 2008.
  - http://www.cse-cst.gc.ca/services/ccs/solaris10ext-e.html

Questions?



david@atsec.com
brian.vetter@bluespace.com