



Intrusion Tolerant CORBA: *Beyond Fault Tolerance*

DOC^{sec}2001

Annapolis, MD

March 29, 2001

Brent Whitmore

bwhitmor@nai.com

Copyright 2001 Network Associates, Inc.
All rights reserved.



Agenda

- Stepping Beyond Fault Tolerance
- Objectives & Approach
- Tolerating Intrusion
- Intrusion Tolerant CORBA Architecture
- So Far, So Good
- Q & A

Stepping Beyond Fault Tolerance

March 26, 2001



@ DOC^{sec}2001

Copyright 2001 Network Associates, Inc. All rights reserved.

Fault Tolerant Systems

- Rely upon system's ability to detect or mask faults
- Can be circumvented by crafty malware
- Examples:
 - Object spoofing
 - Two-faced behavior

Intrusion Tolerant Systems

- *Step beyond fault tolerance.*

- *Step 1:* (Fault Tolerant) Continue to run correctly in the face of *benign* faults
- *Step 2:* (Intrusion Tolerant) Continue to run correctly in the face of *malicious* or *arbitrary* faults
- Arbitrary, potentially malicious executable code termed "*Byzantine*"

Motivation - Why Step Beyond?

- Developing critical applications using CORBA on COTS platforms
- CORBA security
 - Protects at middleware level
 - Applications still vulnerable to O/S and network attacks
- Fault tolerant CORBA does not protect against malicious faults

Objectives & Approach

March 26, 2001



@ DOC^{sec}2001

Copyright 2001 Network Associates, Inc. All rights reserved.

Technical Objectives - Middleware

- Transparently provide intrusion tolerance for CORBA applications
- Eliminate reliance on any single server
 - Secure, reliable group communication directly between clients and replicated servers
- Tolerate Byzantine (arbitrary) faults in servers
- Support heterogeneity (diversity of implementation)

Technical Objectives - Firewalls

- Protocol inspection (of secure reliable multicast)
- Avoid termination of secure connections at firewall
 - Permit end-to-end authentication between clients and servers
- [This work planned for a subsequent phase]

Technical Approach

- Leverage prior work on fault tolerant CORBA; secure, reliable, authenticated multicast; total ordering; Byzantine fault tolerance
- Active replication of clients and servers with voting
- Protect client and server hosts with application proxy firewall; include firewall in secure multicast group
- Integrate with open-source ORB
 - Detect value faults *in middleware*
 - Replace transport layer with secure, reliable, authenticated multicast

Approach — What's Different ?

- All servers are equal
 - Eliminate need for “primary” or “lead” server
- Detect value faults using the ORB
 - Others intercept the transport layer below ORB
 - Encoding of CORBA messages depends on the source platform (i.e, byte ordering)
 - Permit heterogeneous implementations
 - Different O/S, hardware, ORBs
 - Permit parametric comparisons
 - Exact matches not required for inexact values (e.g., floating point)

Approach -- What's Different ?

- Transparent object replication *
- Application proxy firewall integrated into the architecture
 - Better protection for COTS client and server hosts
 - End-to-end authentication of client and server
 - May have better performance than IIOP/SSL proxies

Tolerating Intrusion

March 26, 2001



@ DOC^{sec}2001

Copyright 2001 Network Associates, Inc. All rights reserved.

Tolerating Byzantine Behavior

- Problem first described in *Byzantine Generals* paper (Lamport, Shostak, Pease - July 1982)
- Problem:
 - Commanding general verbally sends orders to $(n - 1)$ lieutenant generals
 - All loyal lieutenants obey the same set of orders
 - Every loyal lieutenant obeys the orders of a loyal commander
 - Loyal generals attempt to reach correct agreement in spite of the traitorous actions of others
- Conclusions:
 - No solution works unless at least $2/3 + 1$ of generals are loyal
 - Voting among generals on each order solves problem after quorum condition $(2/3 + 1)$ met

Tolerating Byzantine Behavior in Distributed Object Systems

- Replicate deterministic objects
- Run replicas in parallel
 - Same inputs
 - Maintain same internal state in all replicas
- Vote on replica outputs before use
 - Output consumer waits for a quorum ($2f + 1$)
 - Consumer tallies votes
 - Accepts “majority” output of the group of replicas
 - Discard subsequent messages for this request/reply
- “Active replication with voting”

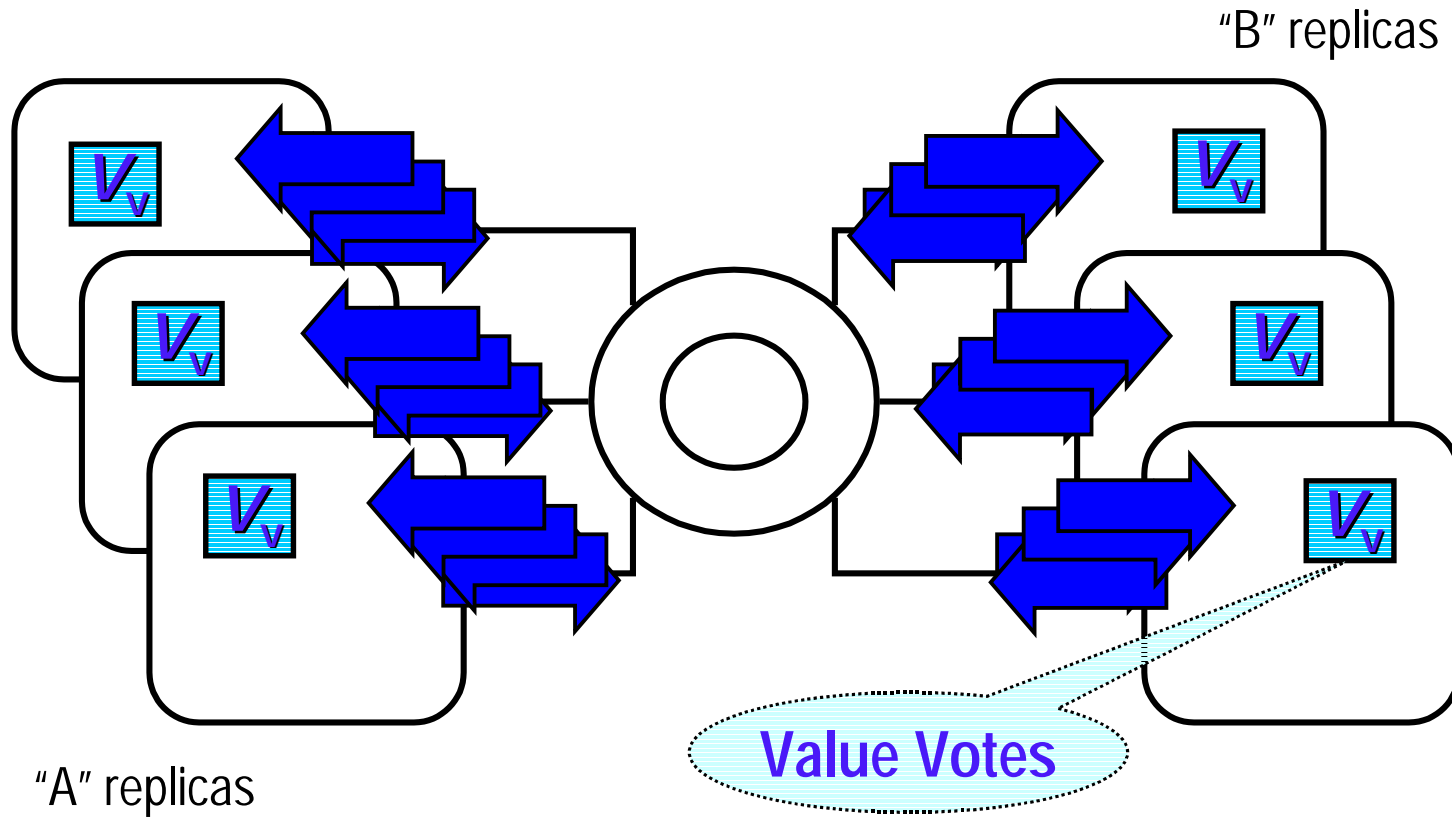
Voting - "A" calls "B"

1 - Send request

2 - Receive request

3 - Send reply

4 - Receive reply



Intrusion Tolerant CORBA Architecture

March 26, 2001



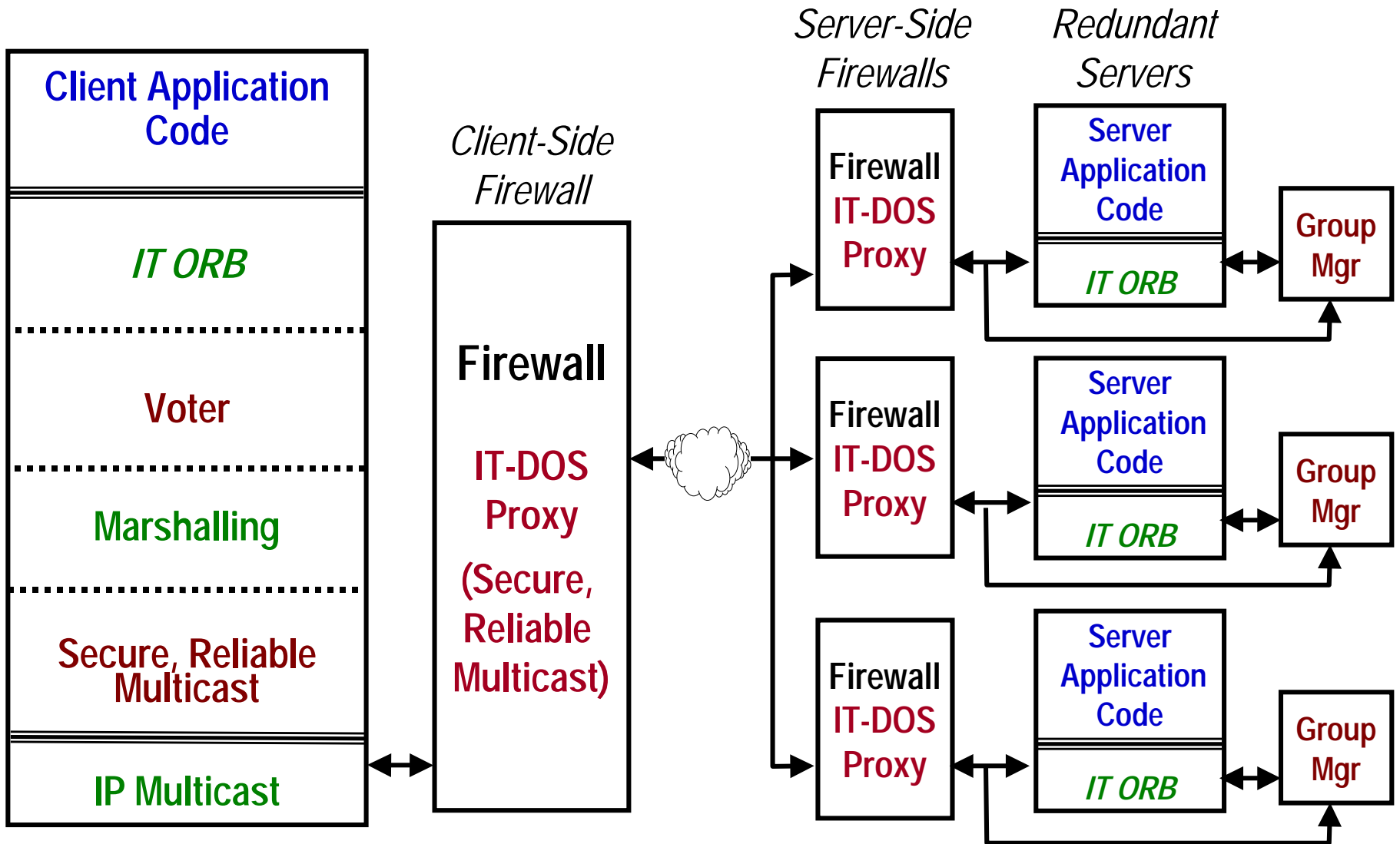
@ DOC^{sec}2001

Copyright 2001 Network Associates, Inc. All rights reserved.

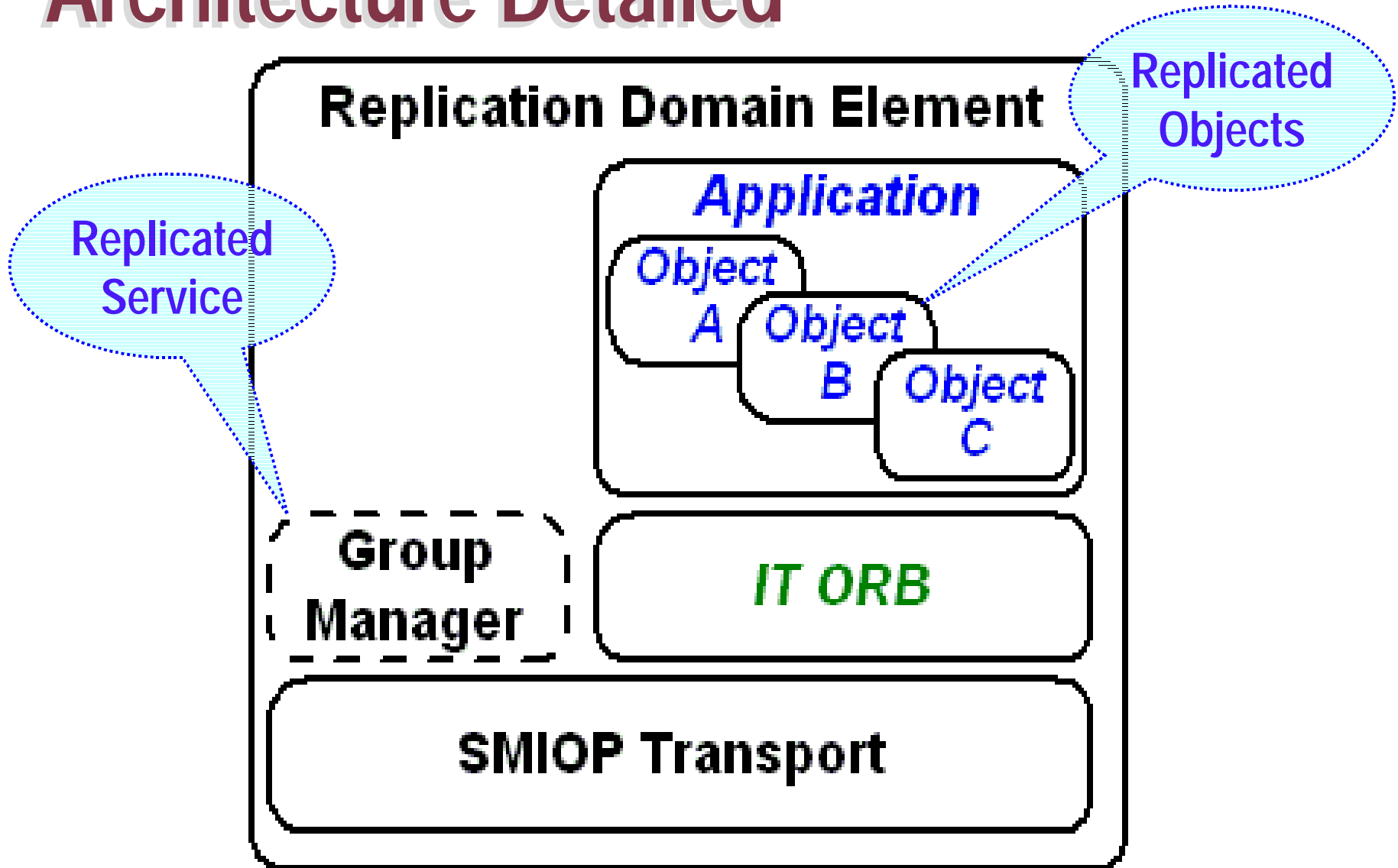
Simplifying Assumptions

- Addition of replacement servers not addressed (first iteration)
- Network does not partition
- Secure configuration mechanism

Conceptual Overview



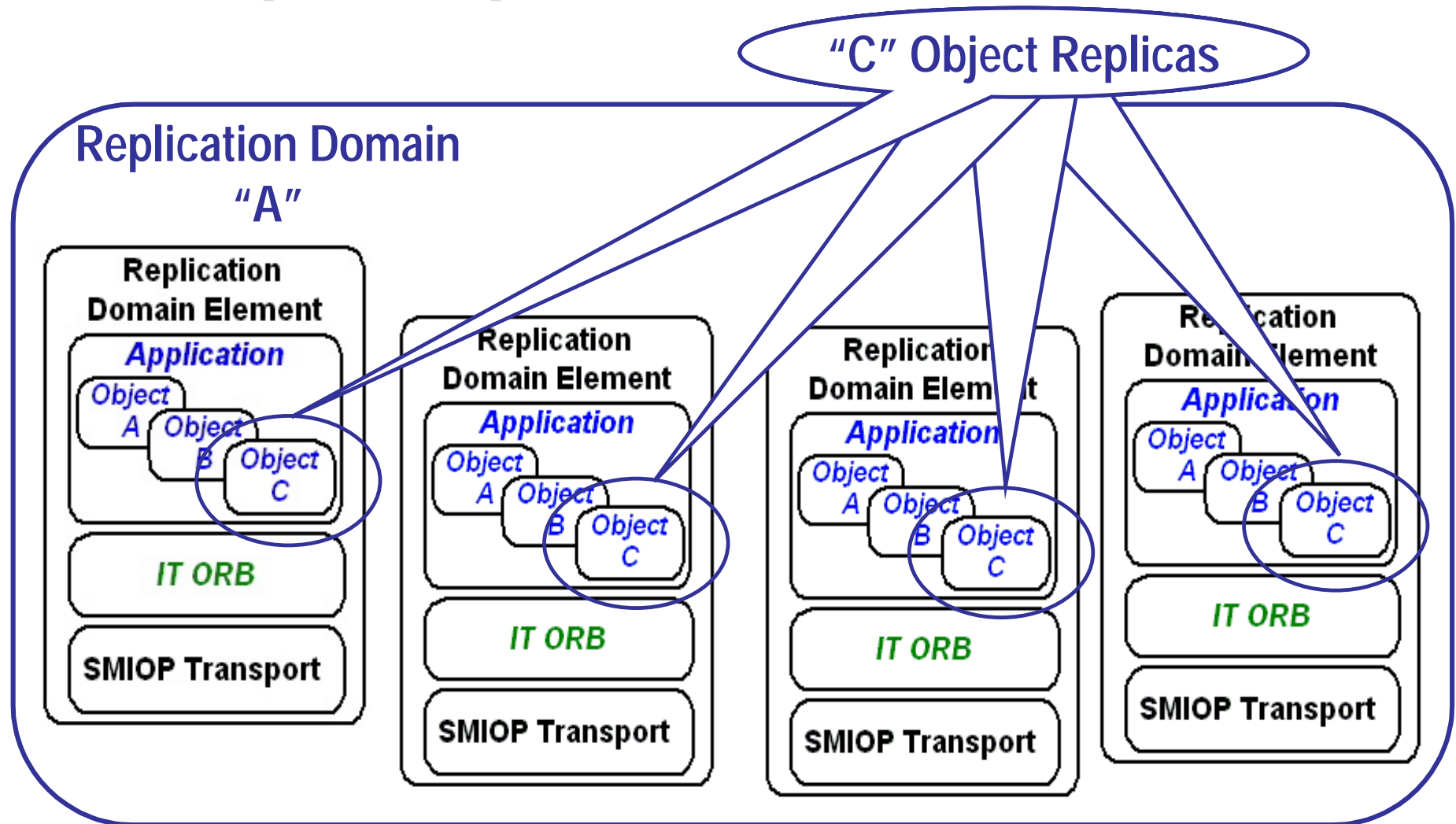
Architecture Detailed



Replication Domains and Their Elements

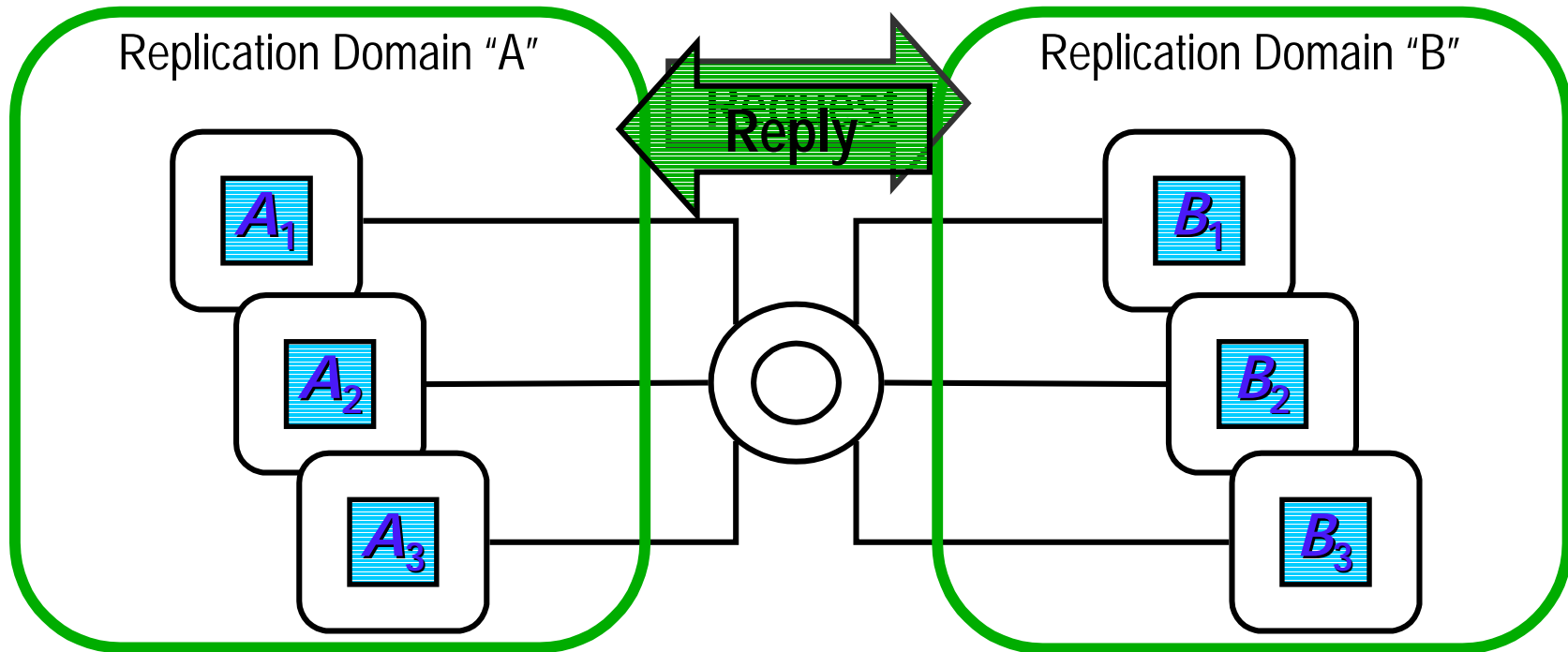
- Replication domains
 - Contain replication domain elements
- Replicated domain element (RDE)
 - Contains application objects
 - Fundamental unit of replication
 - Each RDE in a domain replicates the same set of application objects
 - Authenticates messages sent to others

Example Replication Domain

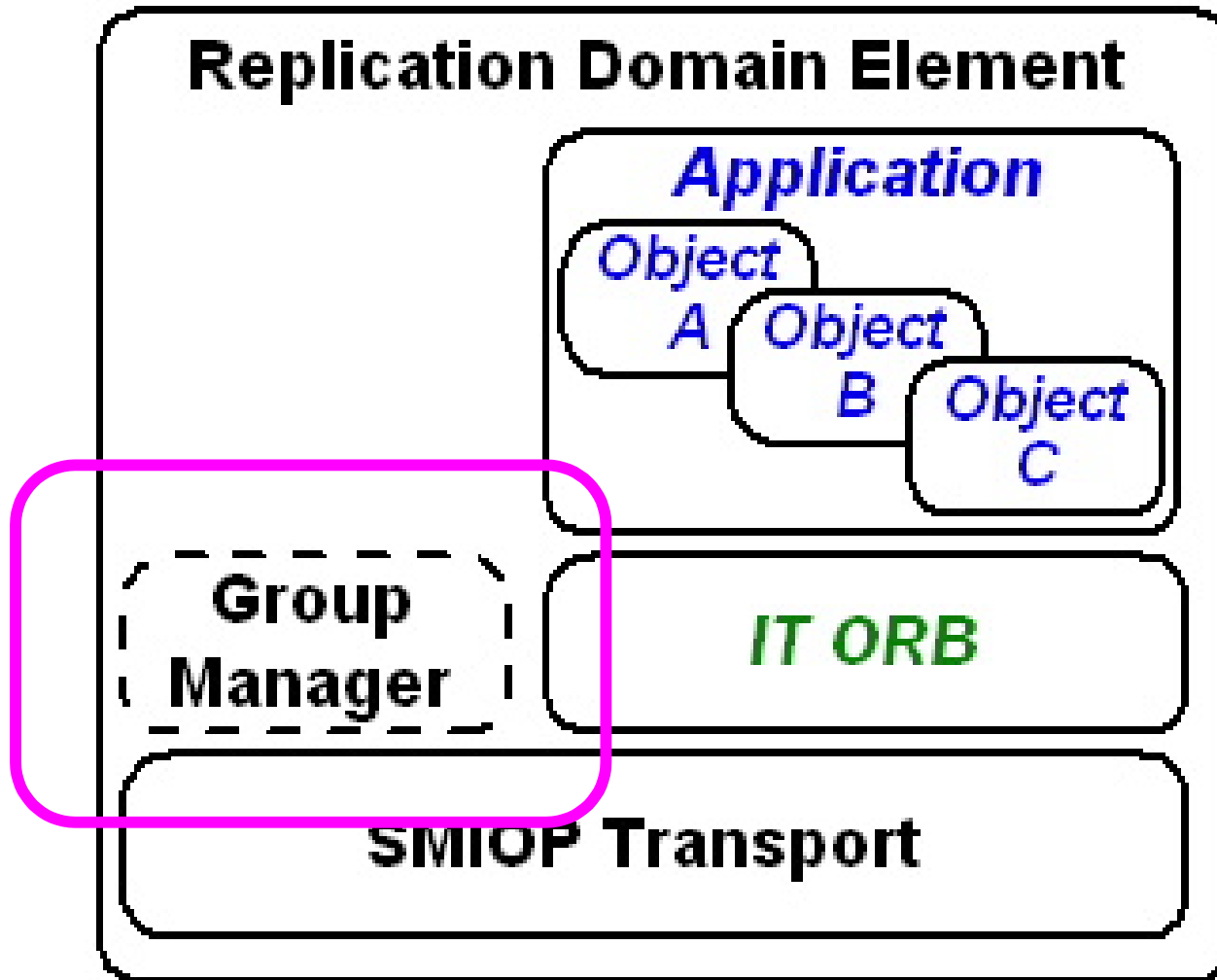


Communication Groups

- Analogous to unicast connections
- Two multicast addresses - one each way
- Secrecy key
- Typically pooled & reused for later invocations



Group Manager Service



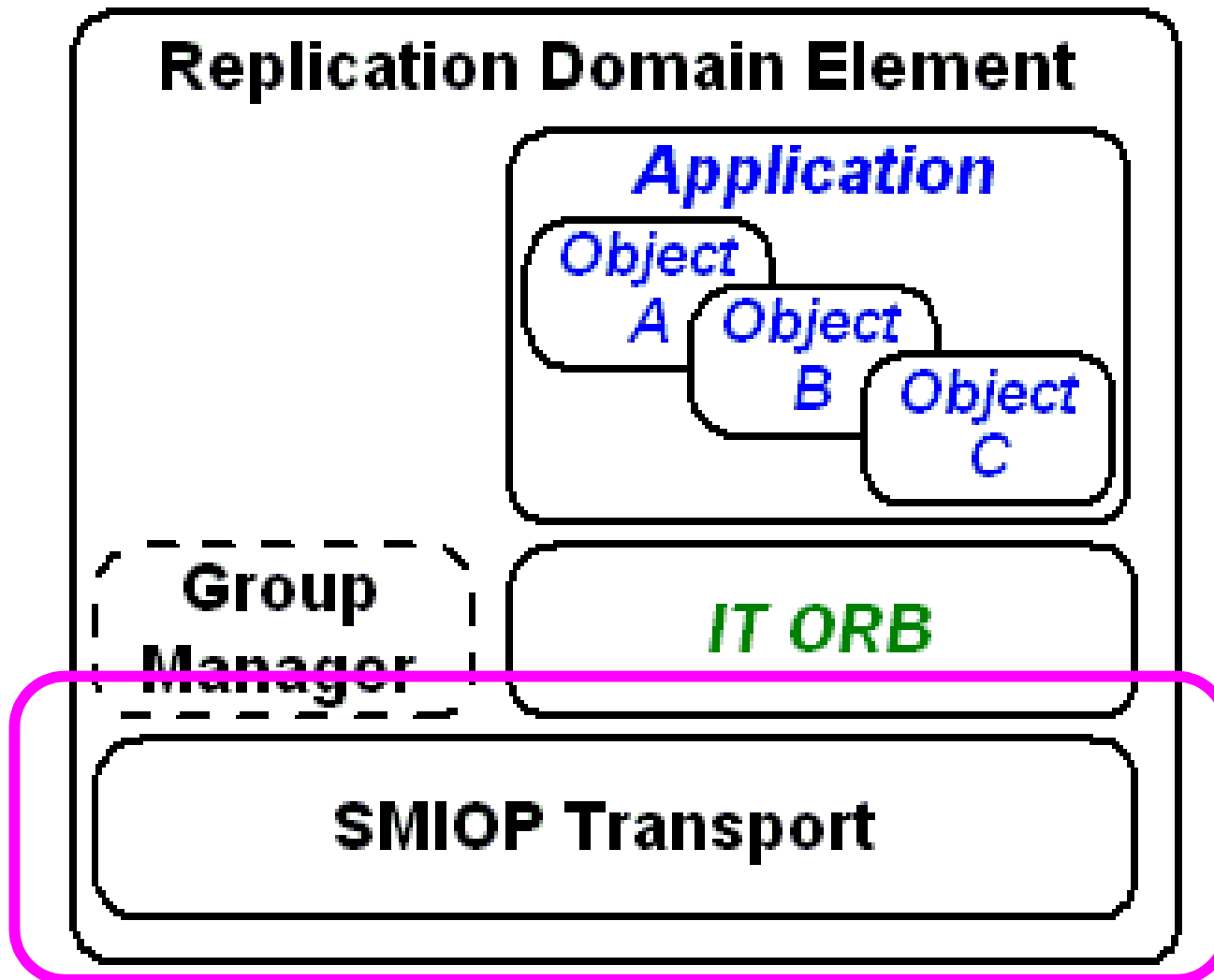
Group Manager

- Replicated service
 - Not an object
 - Uses SRMP directly, no ORB marshalling
 - Not present in every RDE
- Establishes own replication domain
- Located at a well-known multicast address

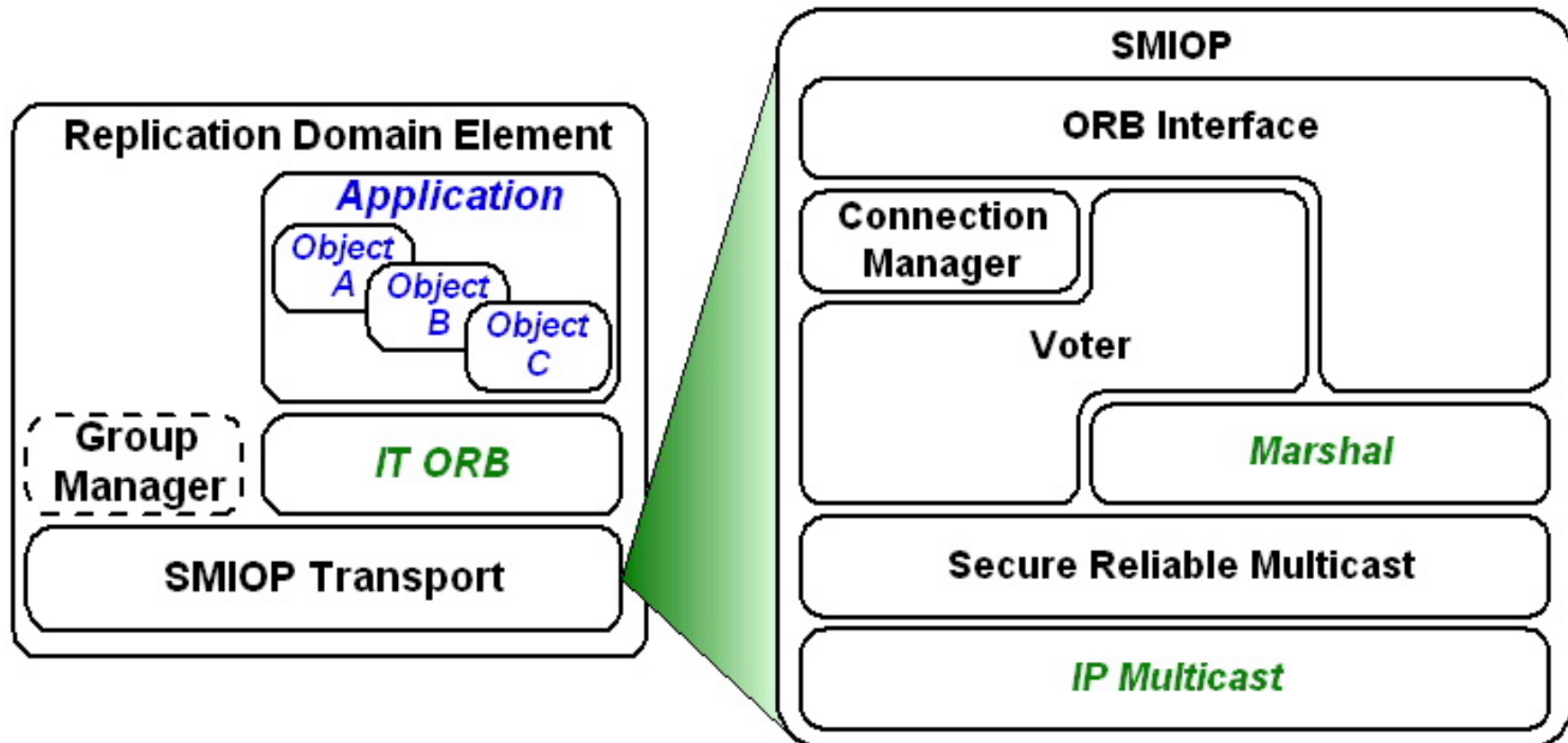
Group Manager Functions

- **Manages groups by:**
 - Assigning multicast addresses to replication domains
 - Generating, assigning & distributing communication group secrecy keys
 - Tracking and managing membership
 - Re-keying for changes in group membership
- **Administers policy**
 - Secrecy
 - Replication
 - Communication/access control

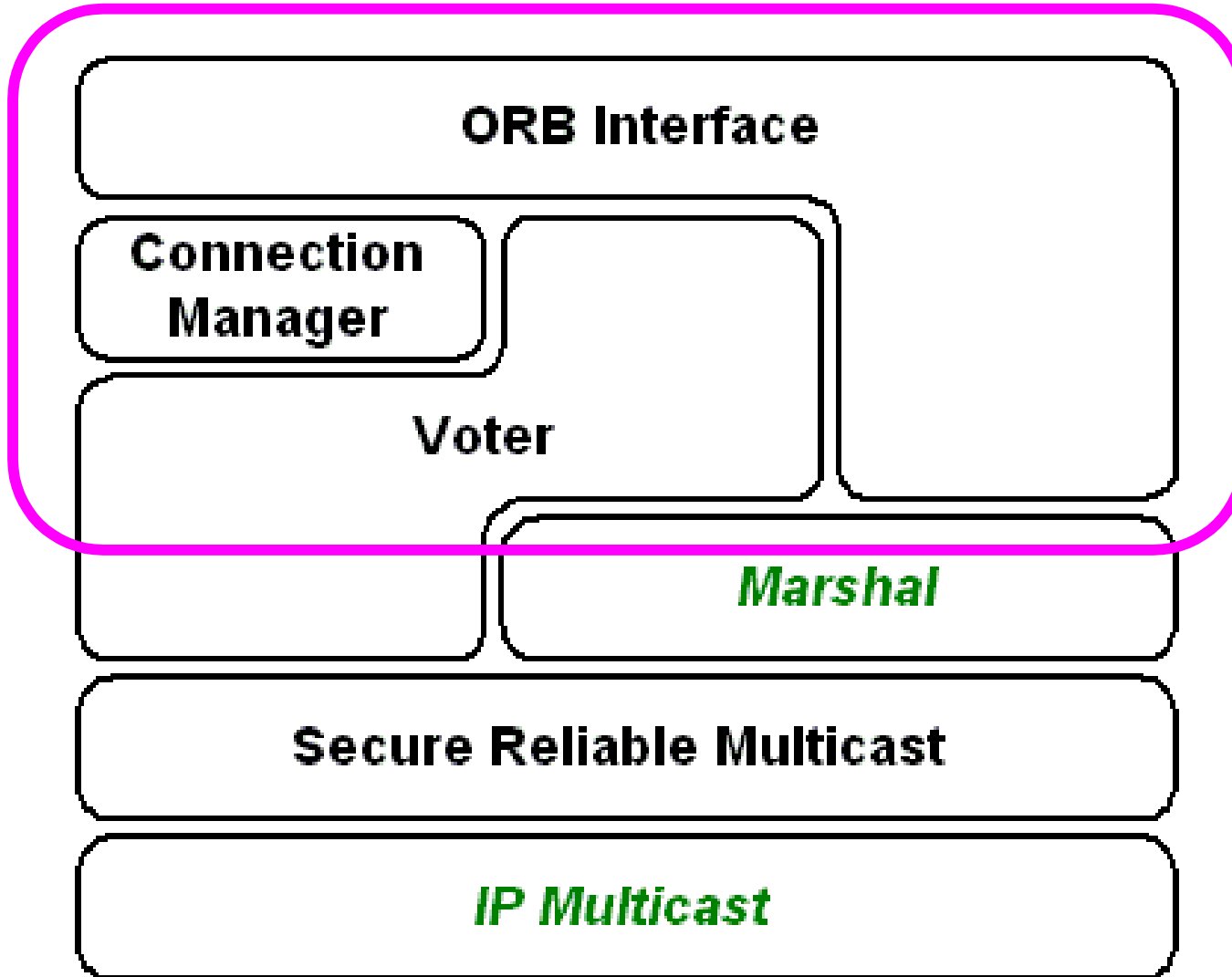
Secure Multicast InterORB Protocol



Secure Multicast InterORB Protocol



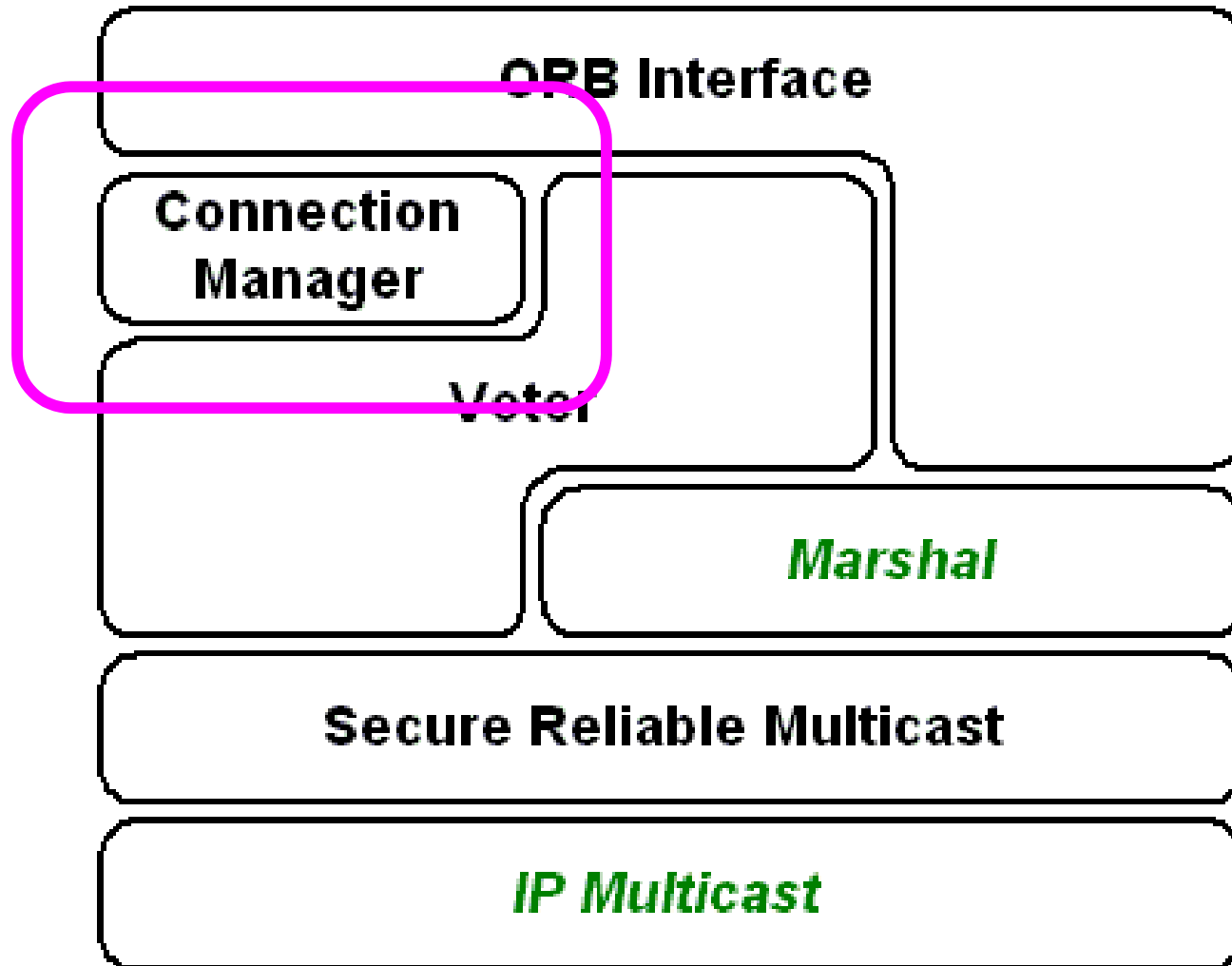
ORB Interface



ORB Interface

- Provides connection-oriented communication to ORB
- Calls marshalling layer directly for outbound messages
- Receives unmarshalled, consolidated inbound messages from voter
- Interfaces supporting installation of SMIOP as an extensible transport
 - Extensible transport extended with callbacks to ORB's marshalling functions from transport

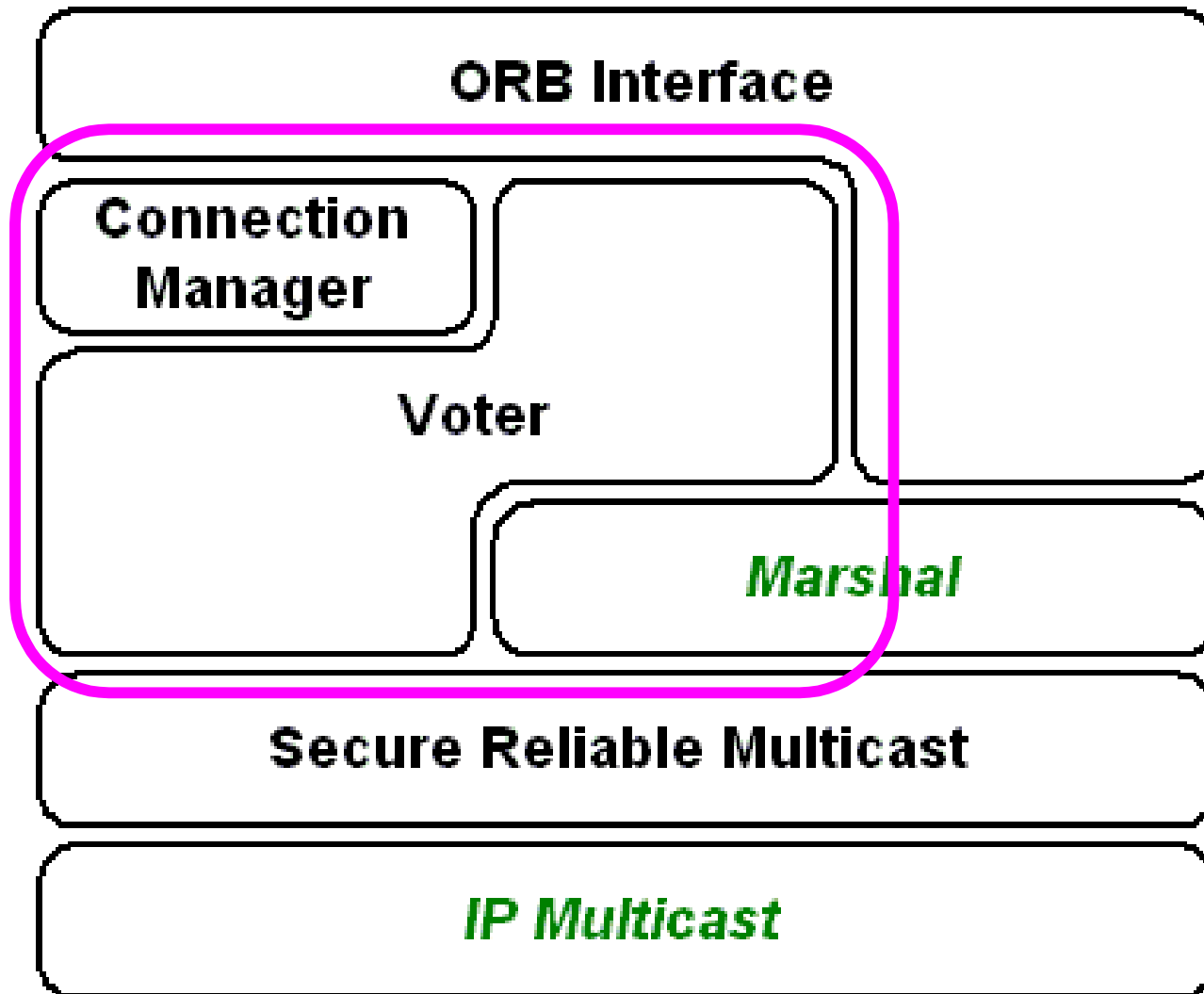
Connection Manager



Connection Manager

- Endpoint for SMIOp connection management messages
- Handles details of open/close/control connection
- Unifies error & timeout indication

Voter



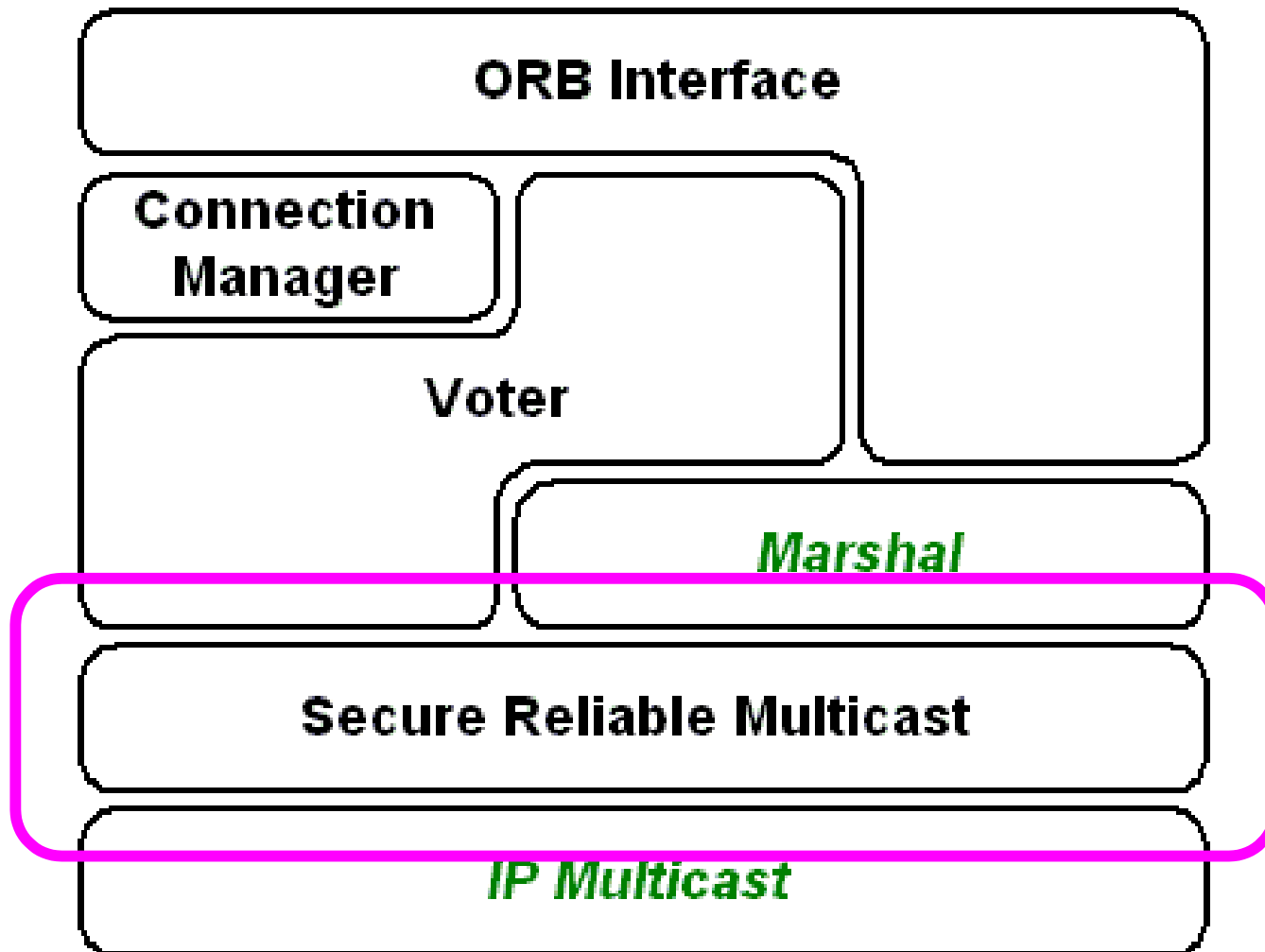
Voters

- **Coalesce duplicate messages**
 - Duplicate invocations from replicated client group
 - Duplicate replies from server group
- **Value voting**
 - Generate agreement on the value of request or reply
 - Identifies objects that disagree as Byzantine
- **Handles Byzantine objects**
 - Remove offending object and all other objects on the same host from all groups
- **Uses Virtual Voting Machine technology**
 - Washington State University/Verizon-BBN work

Voting Machine

- Abstracts out the voting algorithm
- Determine result from possibly inexact matches
- Permits alternatives to majority voting & strict equivalence
- Examples:
 - Mean, median, mode
 - Majority rule
 - Floating point comparison

Secure Reliable Multicast Protocol



Secure Reliable Multicast Protocol

- Uses IP multicast
- Reliable, authenticated transport
- Confidential delivery
- Based on Castro-Liskov's protocol
 - *“Practical Byzantine Fault Tolerance”* — Feb. 1999 & Nov. 2000 (thesis)

SRMP Properties

- **Reliable delivery by processes**
 - Integrity - delivered only once & only if sent
 - Agreement - all delivered or none
 - Validity - delivered if sent
- **Totally ordered**
- **Secure**
 - Source authenticity
 - Message integrity
 - Confidentiality

So Far, So Good

March 26, 2001



@ DOC*sec*2001

Copyright 2001 Network Associates, Inc. All rights reserved.

Expected Achievements

- At least one implementation of an ORB on two or more heterogeneous platforms that tolerates Byzantine faults
- Integrated application proxy firewall support to protect COTS client and server hosts
- Understand trade-off between performance and degrees of intrusion tolerance
- Identify assumptions and residual risks

Progress to Date

- **Developing intrusion tolerant CORBA architecture:**
 - Documented low-level use cases
 - Reviewed prior work
 - Draft architecture paper near completion
- **Selected TAO as the prototype implementation ORB**
- **Received source code for Castro-Liskov group communication system**

Progress to Date

- Development platforms in place
 - PC/Linux/VMWare
 - Sun/Solaris
- Coding imminent
- Firewall support in next phase

Q & A

● Questions?

● Thanks!

● Contact information:

- bwhitmor@nai.com
- <http://www.pgp.com/research/nailabs>