Survivability Through Customization and Adaptability: The Cactus Approach

Matti A. Hiltunen, Richard D. Schlichting*, Carlos A. Ugarte, and Gary T. Wong

Department of Computer Science
The University of Arizona

*Current address:
AT&T Research, Florham Park, NJ

http://www.cs.arizona.edu/cactus/

Work supported in part by DARPA under grant N66001-97-C-8518.

Work done in collaboration with Gregg Townsend and current and former graduate students.

Introduction

Survivable systems:

- Complete missions in time despite failures and attacks.
- Build on security, fault tolerance, safety, etc.
- Require techniques to protect, detect, react, and recover.

Cactus: A framework for constructing configurable and adaptive distributed services and protocols.

Theme: Application of Cactus and its techniques to issues in survivability.

Fundamental techniques:

- Fine-grain customization through configurability.
- Dynamic adaptation.

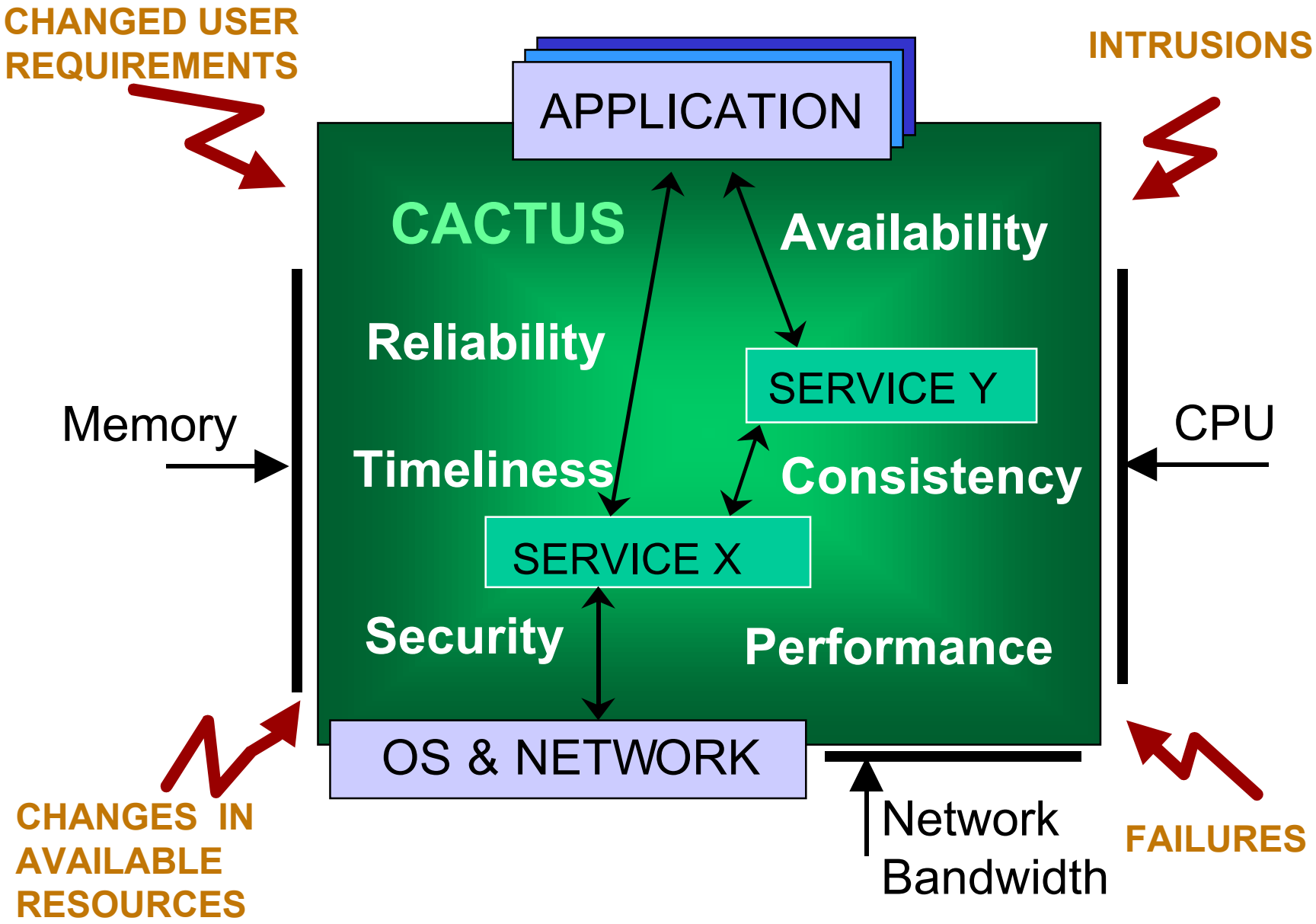
Advantages:

- Customizable cost versus protection.
- Customization for scale.
- Artificial diversity through configuration.
- New survivability techniques can be easily added.
- Dynamic adaptation to attacks and changes in survivability requirements.

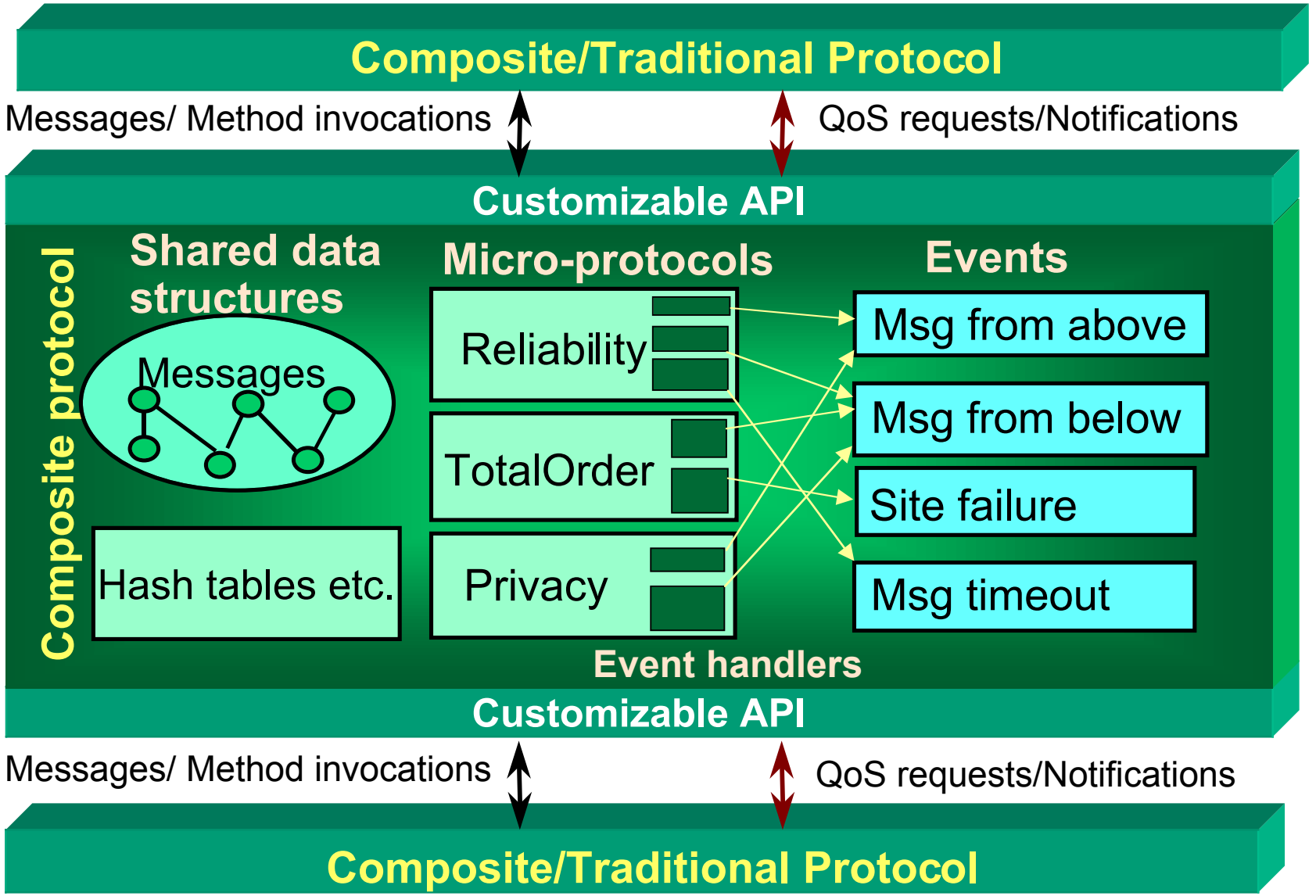
Presentation Outline

- The Cactus Approach.
- Cactus Survivability Mechanisms.
- Services - Current and Planned.
- Status
- Conclusions.

The Cactus Approach



Cactus Model



Configurability in Cactus

Configurability through independence between micro-protocols provided by the Cactus mechanisms:

Events:

- Dynamic binding of event handlers to events.
- Flexible: parameter passing, synchrony, ordering.

Shared session and protocol variables.

Cactus messages.

- Msg header = dynamic set of named msg attributes.
- Attribute scope: LOCAL, PEER, STACK.
- Coordination mechanism for sending msg to next protocol.

Adaptability in Cactus

Idea: Adaptation by dynamically reconfiguring a service.

Examples:

- FT: changing multicast algorithm to accommodate a change in failure model assumption.
- Security: increasing level of encryption to counteract an intruder.

Cactus mechanisms:

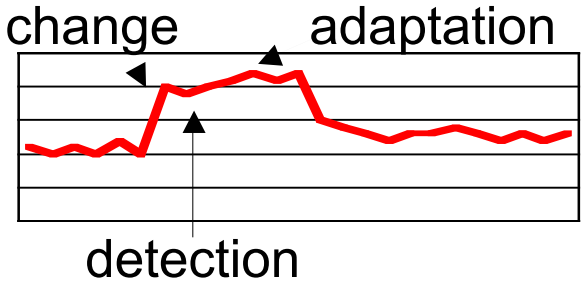
- Activation/deactivation of micro-protocols through event handler binding.
- Dynamic code loading + activation of new code.

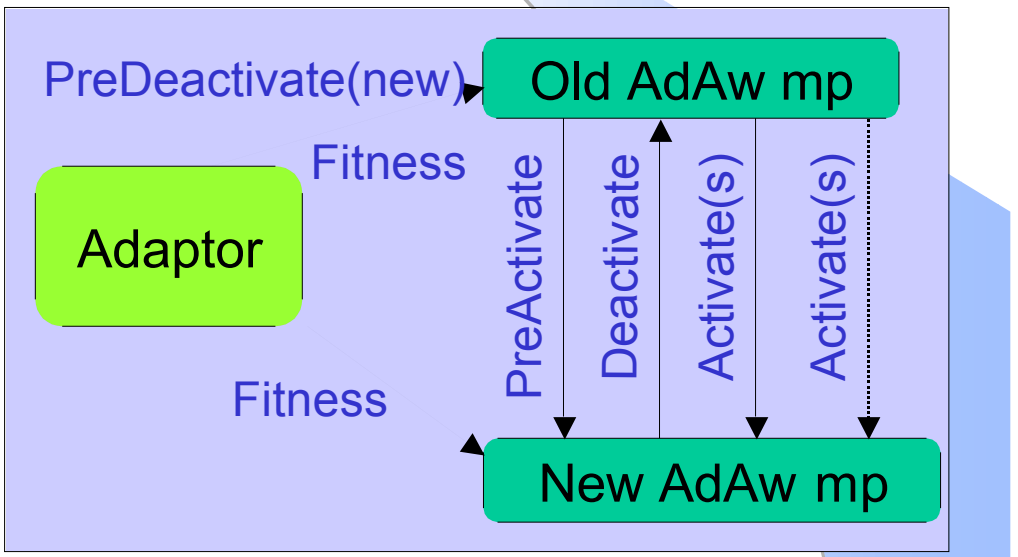
Coordination issues:

- When to activate/deactivate a micro-protocol.
- Distributed coordination of adaptation.

Goal: smooth adaptation.

Approach: a multiphase transition from old to new micro-protocol.





Cactus Survivability Mechanisms

Fault tolerance and security

Fundamental properties for survivable systems.

Different FT and security mechanisms can be implemented as micro-protocols.

FT: retransmission, atomicity, checksums, object/process replication, message logging.

Security: cryptographic methods for privacy, authenticity, integrity, replay prevention, non-repudiation.

Implementation options:

1. Integration with an existing configurable service (e.g., communication or file service) or
2. Separate fault-tolerance or security service.

Artificial diversity

Harder for an intruder to apply same attack method on different installations of a system/service.

Configurability a mechanism for artificial diversity:

- Natural diversity through customization for user requirements and characteristics of the execution environment.
- Additional diversity by providing alternative micro-protocols with same service property (e.g., different encryption algorithms for privacy).

Adaptability

Survivable systems exhibit adaptive behavior:

- Automated reactions to intrusions, system state restoration.
- Service upgrades to handle new attacks.
- Dynamic change of security level when intrusion suspected.

Current work: performance and fault-tolerance adaptations.

Future work: security and real-time adaptations.

Challenges:

- Adaptation mechanisms must be intrusion tolerant (e.g., message authentication, Byzantine methods).
- Adaptation must happen in bounded time.

Transparent survivability

Legacy and off-the-shelf applications not often survivable enough

⇨ transparent enhancement of survivability necessary.

Replacement of underlying communication/OS services or transparent insertion of middleware services:

- Linux loadable kernel modules (Cactus comm. protocols).
- Interception of signals on Linux and Solaris (Cactus DSM service).
- Smart stubs, interceptors, and DSI on CORBA allow insertion of Cactus services between an application and the ORB.

Current Cactus Services

A number of distributed services implemented to validate and demonstrate the Cactus approach.

Examples:

- RTD Channels (real-time, reliability, ordering).
- Group membership (ordering, consistency).
- Distributed shared memory (consistency, replacement, etc).
- Secure communication service.
- Group RPC.
- System monitoring service.

Focus on basic attributes rather than adaptation.

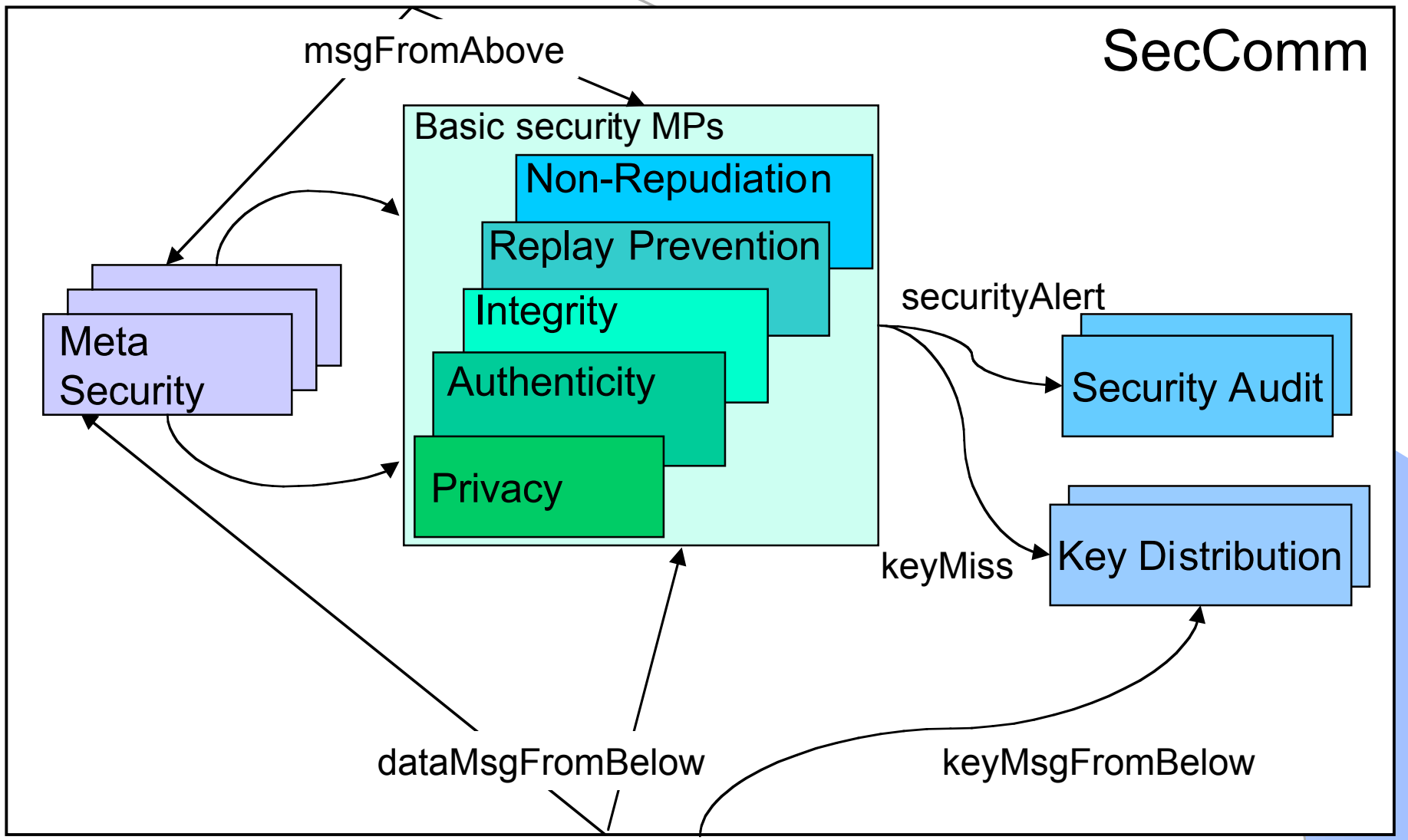
Secure Communication Service

Increasing emphasis on customizing communication security (e.g., IPSec, SSL, TLS).

SecComm: customizable secure communication service implemented using Cactus:

- Multiple basic security MPs for privacy, integrity, authenticity, non-repudiation, replay prevention, key distribution, etc.
- Meta security MPs: use basic security MPs to construct more complex protocols, e.g., alternating encryption.
- MPs simple $\Rightarrow$ easy to add custom security MPs.
- Arbitrary number and combinations of the micro-protocols allowed $\Rightarrow$ arbitrarily high security at arbitrarily high cost.

MP classes and event interactions



SecComm performance

Implemented on a cluster of Pentium PCs (133 MHz) running MK 7.3 OS from Open Group/RI connected by a 10 Mbps Ethernet.

micro-protocols	roundtrip
none	3.59 ms
XOR	3.82 ms
DES	6.75 ms
DES, XOR	6.96 ms
DES, XOR, Blowfish	8.98 ms
MD5	4.01 ms
SHA	3.99 ms
MD5, SHA	4.36 ms

Package size: 100 bytes.

Key lengths:

- DES 56,
- Blowfish 448,
- XOR 64

Average IP roundtrip time:

- 3.03 ms.

Group RPC

Replicated RPC to a group of servers.

Novel feature: customizable failure model (crash, send/receive omission, late/early timing, value, Byzantine).

Other properties: synchronous/asynchronous, FIFO/total order, atomicity.

11 micro-protocols, dozens of configurations.

clients	servers	failure model	fifo	total
1	1	none	3.3 ms	3.6 ms
1	2	crash	4.2 ms	6.2 ms
1	2	send omission	4.1 ms	6.6 ms
1	3	rec. omission	5.3 ms	10.5 ms
1	3	byzantine	2181 ms	18924 ms

System Monitoring Service

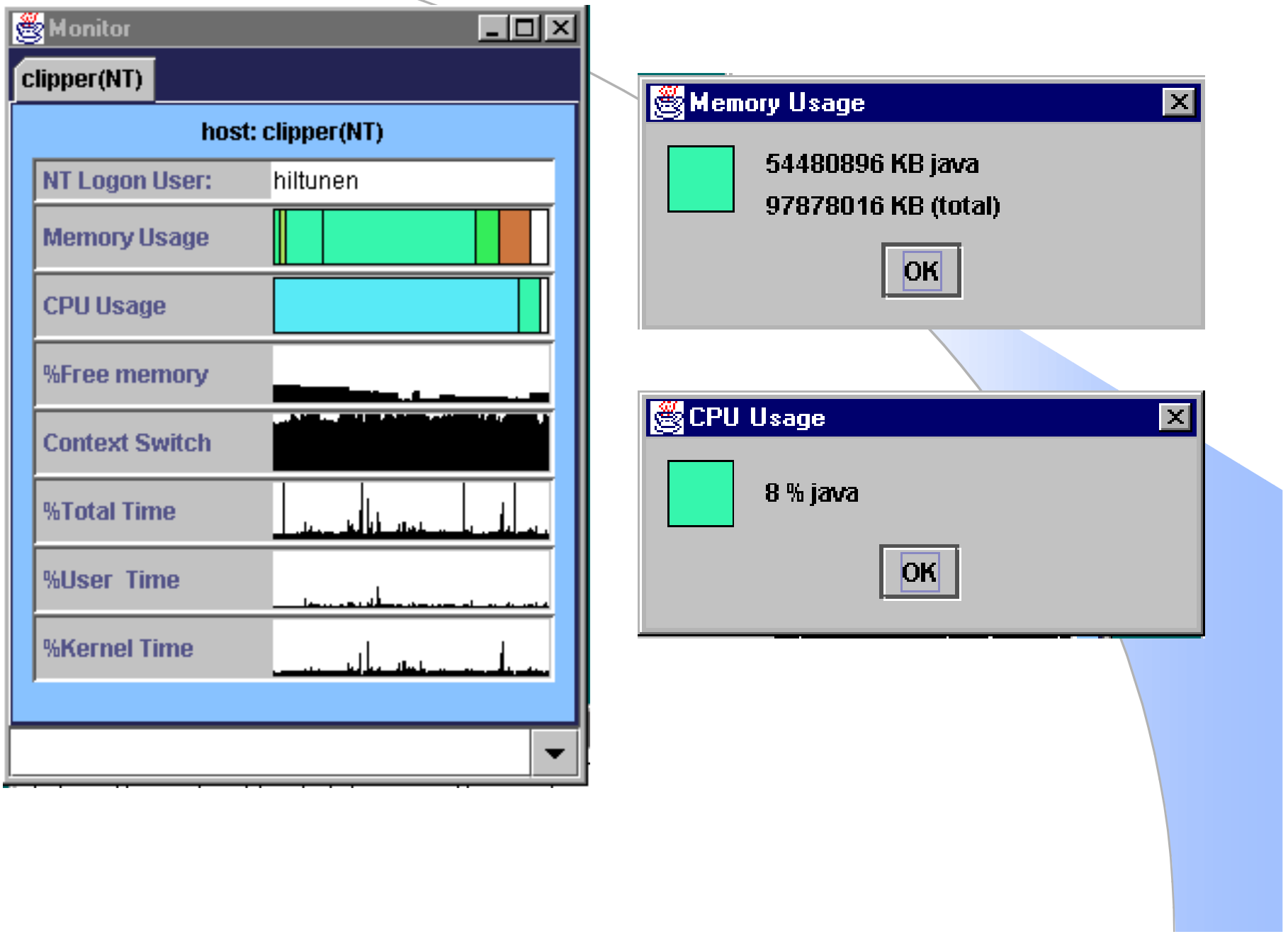
A dynamically configurable distributed system monitor for NT and Linux implemented using Cactus/J 2.0 (Java).

Each aspect of system monitoring implemented as a separate micro-protocol:

- Users.
- Processes: CPU/memory usage, etc.
- Processor: Available memory, context switches, etc.

Micro-protocols can be loaded/unloaded at runtime.

New features could be easily added, e.g., monitoring for survivability.


Monitor
clipper(NT)
host: clipper(NT)
NT Logon User:
hiltunen
Memory Usage
CPU Usage
%Free memory
Context Switch
%Total Time
%User Time
%Kernel Time
Memory Usage
54480896 KB java
97878016 KB (total)
OK
CPU Usage
8 % java
OK

Services for Survivability

Cactus mechanisms could be used to construct services specifically geared for survivability.

Intrusion detection.

- Extensible: new data collection and analysis micro-protocols.
- Customizable coverage versus performance/resource utilization/inconvenience.
- Customization for current mode of operation.
- Short term adaptation to detected threats.
- Long term adaptation: integration of new techniques.

Survivable data storage.

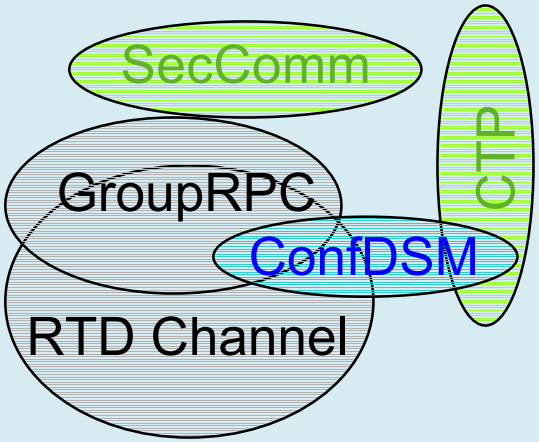
- Confidentiality: cryptography, data fragmentation.
- Integrity for intrusion detection.
- Replication for availability.
- Checkpointing, change logging for recovery.

Access control and authentication.

- Customized authentication => diversity.
- Adaptive authentication.

Status: Prototypes and Services

	Solaris	MK	NT	Linux	In-kernel Linux
Cactus 1.0	C++	C		C++	
Cactus 2.0	Java	C	Java	Java C C++	C

QoS Attributes	Example services
Security Fault tolerance Consistency Real time	

System level	Services
Application "Middleware" Protocols	DistMonServ ConfDSM ConfCORBA GroupRPC Membership SecComm CTP RTD Channel

Colors:
- completed, released,
- in progress, planned.

Conclusions

Configurability and adaptability supported by the Cactus platform important mechanisms for survivable systems.

- Customization of tradeoffs.
- Extensibility to introduce new survivability techniques.
- Adaptation to attacks.

Cactus framework and example services available through Cactus home page: http://www.cs.arizona.edu/cactus/