Autonomic Multi-Agents Security System for multi-layered distributed architectures

Christian Contreras
Agenda

• Introduction
• Multi-layered distributed architecture
• Autonomic computing system
• Multi-Agent System (MAS)
• Autonomic Multi Agent Security System
• JADE Framework
• Proposed solution features
• Conclusions
• Comments and questions
Introduction

• Multi-layered and distributed system architecture
• Characteristics: heterogeneity and complexity.
• Increase Cyber threats. → Security concerns grows.
• Maintain the SLO or SLA.
• We need security systems with a new approach.
• System that can respond to current threats and future threats in a quicker and efficient way.
• Autonomic Computing System (ACS) + Multi-agent System (MAS)
  – Autonomous system
  – Self adaptive
  – Self protecting
  – Knowledge Base → learning process
Multi-layered distributed architecture
Architecture comparison

• Autonomic Computing Properties
  – Self-Configuring
  – Self-Optimizing
  – Self-Healing
  – Self-protecting

• MAPE-K Framework
  – Define behavior

• Multi-Agent System Properties
  – Autonomy
  – Reactivity
  – Pro-activeness
  – Social ability

• Desire Belief Intention (BDI) Model
  – Define agent behavior
Autonomic Computing Systems: MAPE-K framework
Autonomic Computing Systems: MAPE-K framework

- **Monitor** and collects the details from the managed resources using sensors.
- **Analysis** perform complex data analysis. It uses the stored knowledge data.
- **Plan** structures the actions needed to achieve goals and objectives.
- **Execute** changes the behavior of the managed resource using effectors.
- **Knowledge** standard data shared that includes data such as topology information, historical logs, metrics, symptoms and policies.
Autonomic Computing System: Basic Properties

Increase Responsiveness
Adapt to dynamically changing environments

Operational Efficiency
Tune resources and balance workloads to maximize use of IT resources

Self-Configuring
Self-Optimizing
Self-Healing
Self-Protecting

Business Resiliency
Discover, diagnose, and act to prevent disruptions

Secure Information and Resources
Anticipate, detect, identify, and protect against attacks
Multi Agent System properties

• **Autonomy** is the freedom that an agent has to make decisions.

• **Reactivity** is the response to changes in the environment.

• **Pro-activeness** is the exhibitions of a goal-directed behavior.

• **Social ability** is the ability to successfully interact and cooperate with others agents.
BDI behavior model agents

- Desires (Goals)
- Beliefs (Knowledge Base)
- Intenurons (Plan Instance)
- Plan Library (Procedural Knowledge)
- Interpreter (Reasoner)
- Event Queue
- Intention Queue
- Sensor Input
- Environment
- Action Output
Multi Agent System Architecture
Multi Agent System Architecture: Type of agents

• Reactive agent
  – It reacts to an event and does not perform any kind of deduction.
  – It always behaves in the same way. (pre-defined)

• Practical reasoning agent
  – It has a proactive behavior and make decisions on its own.
  – First it deliberates what state of affairs to achieve
  – Second it decides how to achieve the desired state of affairs.
  – Predefined sets of actions. (Plans)

• Deductive reasoning agent
  – There is a description of the environment.
  – There are a set of rules.
  – The agent deduces the appropriate set of actions
Autonomic Multi-Agent Security System

[Diagram showing the components of an autonomic security system, including Desires (Goals), Change Request, Plan Library, Plan, and Intention Queue.]
Autonomic Multi-Agent Security System
JAVA Agent DEvelopment (JADE) Framework
Autonomic Multi-Agent Security System

JAVA Agent DEvelopment (JADE) Framework

Multi-agent distributed application

Agent  Agent  Agent  Agent  Agent  Agent

JADE LAYER

Container  Container  Container  Container

JADE

JAVA VM LAYER

J2SE  J2EE  PersonalJava  CLDC

Internet  Wireless environment
JADE Architecture

- **Container**: a running instance of the JADE runtime and may contain several agents (zero or more agents).
- **Platform**: the set of all active containers
- **Main container**:
  - First container to start in a platform
  - Must always be active
  - All other containers (normal, non-main) must register with host and port of the main container.
- **Special agents**
  - Agent Management System (AMS).
    - Provides the naming service
    - Represents the authority
  - Directory Facilitator (DF).
    - Provides a Yellow Pages service
- **Communication**: is based on an asynchronous message passing paradigm (**ACL** language).
Autonomic Multi-Agent Security System
Autonomic Multi-Agent Security System

Cloud

Multi-layer

Middleware

Cross layer security

Resources
Autonomic Multi-Agent Security System

• Multi-layered distributed system.
• Cross Layer security.
• Multi agents behave emulating the MAPE-K framework.
• Modeling the Security Knowledge Base with ontologies.
  – Security policies, rules, threat signatures, etc.
• Supervised and unsupervised learning techniques to acquire new knowledge.
• Agents are light-weight process.
• Detect and react faster.
Autonomic Multi-Agent Security System

• JAVA Agent DEvelopment (JADE) Framework.
  – A fully distributed system inhabited by agents, each running as a separate thread, potentially on different remote machines, and capable of transparently communicating with one another.
  – Full compliance with the FIPA specifications. (Foundation for Intelligent Physical Agents)
  – Uses FIPA Agent Communication Language (ACL).
  – Efficient transport of asynchronous messages via a location-transparent API.
  – Support agent mobility.
  – Support ontologies and content languages
Autonomic Multi-Agent Security System

• Self-Protecting ➔ Secure Information and Resources
• Prediction
  – Supervised and unsupervised techniques.
• Detection and Inform
  – Monitor performance system metric, request analysis, log analysis, system corruption, data corruption, IDS.
  – Build security traps.
• Identify
  – Attack signatures, system vulnerabilities, bottlenecks, congestion.
• Protection
  – Change cryptographic system, communication protocols, ACM, flow control, etc., filter requests, drop requests, limit resource usage, limit resource access, block resources, etc.
Autonomic Multi-Agent Security System

• Agency trust model
  – Security policies.
  – Registration service
  – Supervise behavior
  – Upgrade process

• Agents autonomic security schema
  – Self-protecting
  – Self-healing
  – Self-configuring
  – Self-optimization
Conclusion

• Autonomic Multi-Agent Security System for multi-layered distributed architectures is a defense in depth approach.

• Agent Autonomic Security System that can self-protecting, self-healing, self-configuring, self-optimization itself.

• Knowledge base using ontologies can provide
  – Adaptability for future threats.
  – Reach and share knowledge across the web.

• JADE framework provides a solid platform with potential grow. → Jena Apache Ontology API.

• Agents are light-weight processes that can accomplish scalability and portability.

• Provide support to maintain the SLA or SLO.
Comments and Questions