PARTIAL COMPUTER HOMEOSTASIS THROUGH SYSLOG ANALYSIS USING AUTONOMOUS EPISTEMIC AGENTS

Cameron Hughes, Ctest Laboratories
Tracey Hughes, Ctest Laboratories
Trevor Watkins, Kent State University
James Dittrich, Advanced Software Construction

KCAP KNOWLEGE CAPTURE CONFERENCE 2015

The Problem

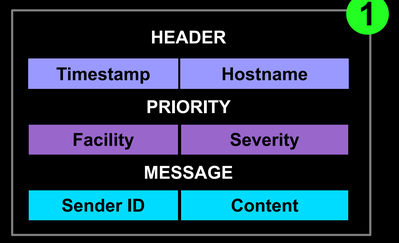
The proliferation of mobile computing, the Internet of Things, hosting services, and cloud computing has increased the burden of computer log file analysis for system administrators, network and security analysts, and large server hosting organizations. The log analysis process monitors and controls the overall health of the computer systems that support these technologies. But voluminous amounts of log entries produced by these technologies has made real-time log analysis by human effort untenable and automated real-time analysis essential.

Our Approach

We describe an approach to partial computer self-regulation that is knowledge-based by using autonomous epistemic agents to perform log analysis within a hybrid deductive-abductive first order logic model. The epistemic agent utilizes *a priori knowledge* of Unix/Linux-based computer systems with *posteriori knowledge* extracted from log messages in order to uncover negative scenarios that reflect problems in the system. The necessary adjustments are made to regulate a computer system's homeostasis.

Log Messages and Analysis

1

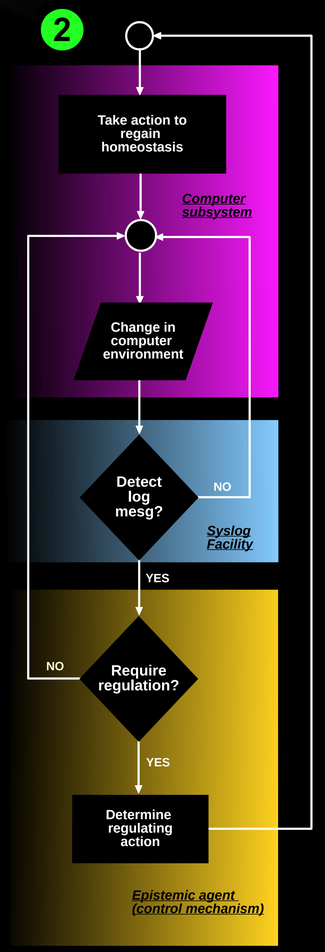


Syslog Request For Comments (RFC) – 5424 describes the standard for a syslog messaging subsystem (1). The **Priority** contains the **Facility** designates what kind of program is logging the message, and the **Severity** level indicates the relative importance of a log message or event. The **Message** contains the **Sender ID** that identifies the source of the message, and the actual message content. The syslog server can collect 100,000s of messages daily and stores them in a syslog log file. A system administrator examines these files by using insight, judgment and rudimentary tools before a diagnosis or information synthesis becomes apparent.

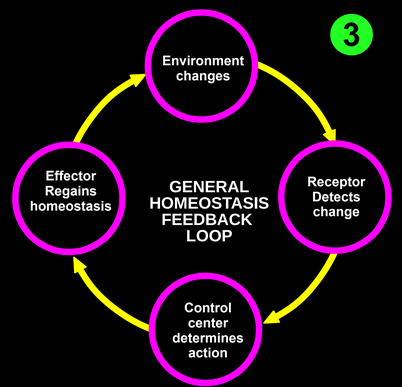
PARTIAL COMPUTER HOMEOSTASIS AND S.O.S. HISTORIES

S.O.S (**System Operational Status**) history cannot be counted on for a complete picture because not all errors and exceptions are logged. TAMI-2 constructs a S.O.S history of a system by analyzing errors, or exceptions identified in syslog that are still reflected in the current state of the system. Based on RBAC permissions, necessary adjustments to contribute to the system's homeostasis are performed.

2



3



4

```
epistemicAgent(HostName,
EvaluationInterval, Logfile)
T := EvaluationInterval
L := set of Logfile  entries
M := emptySet of Model Interpretations
S := set of system maintenance areas
for  ∀Le ∈ (L  ∩  T)
     M.add(modelInterp(Le))    //Q(F,T)
end for
for  ∀x ∈ S
     J  := abduce(x ∩  M)
end for
NS  := negativeScenarios(J)
PS := positiveScenarios(J)
for  ∀s ∈ NS
     State = currentComputerState()
     if  s ⊆ State ∧ rbacPermission(s)
     then regulate(s)
     end if
         syslogReport(s)
```

5

POSIX/LINUX Systems Admin. Areas	Knowledge Source & Ontology Type
File System Maintenance	POSIX ISO/IEC 9945 1003 Vol 4 International Standard
User Account Maintenance	Domain
Memory Management	RFC 5424 definition for syslog
System/Application Process Maintenance	Domain & Application
Commonly Used Computer Peripherals	Linux/Unix man pages
Syslog Files	Domain & Application
	Linux/Unix System administration guides
	Application
	Errno.h, error.h
	Domain

6

Quadrant	Acronym	Description
I	SOSS	Single Occurrence, Single Segment
II	SOMS	Single Occurrence, Multiple Segments
III	MOSS	Multiple Occurrences, Single Segment
IV	MOMS	Multiple Occurrences, Multiple Segments

7

Primary Functions	Descriptions
modelInterpretation	Takes a tokenized log entry and returns FOL formula for the hybrid deductive abductive model generated while creating G_1 for E_S.
abduce	Produces a set of candidate hypothesis that associates log entries with negative or positive scenarios.
negativeScenario	Produces a set of negative scenarios that can be verified from the hypothesis.
positiveScenario	Produces a set of positive scenarios that can be verified from the hypothesis.
rbacPermission	Determines what level of action an agent has the authority to take based on the scenario type.
regulate	Performs the necessary change to the system to contribute to the system's homeostasis.
syslogReport	Reports any actions taken, and the analysis, diagnosis for log entries according to the configuration constraints originally setup for syslog.

8

Tools Used	Purpose	Knowledge Type
COGUI	CG ontology development	A Priori
PROTEGE	KB development	A Priori
PROLOG	Abductive/Deductive Inference Engine	Posteriori

Epistemic Agent's A Priori POSIX Knowledge + Scenarios: G_1

Our epistemic agent has common sense knowledge about a POSIX compliant or Linux operating system environment in 6 areas (5) with a set of negative and positive server scenarios, make up G_1 .

FOL models, ***M = (D, F),*** implement their respective domain and application ontologies.
D is the domain taken from the 6 areas.
F is a hybrid interpretation that combines deductive and abductive methods to establish the semantics of a formula in ***M***.

Negative and Positive Scenarios

Let **T** be defined by the time range $T_1...T_2$. Let any negative scenario or positive scenario in G_1 be a fluent ***f*** then our epistemic agent has the relation:

$\Diamond$ **holdsAt(*f*,T)** $\leftrightarrow$ **(p** $\rightarrow$***f*****)** $\wedge$ **p** $\in$ **J** $\wedge$ **Vc(p) > 0**

The **Positive Scenarios** in G_1 are based on the tuple **<Facility,Severity,Area>** for any log messages present at **T** for a given facility with no *syslog messages and a Severity level >= 5.*

Negative Scenarios are based on the tuple where:

Quadrant analysis **Q(L_e,T)** for all L_e (log entries) where **Q: (L_e,T)** $\rightarrow$ **O** and **O = {SOSS, SOMS, MOSS, MOMS} (6)**, and a semantic analysis of L_e from the agent's posteriori knowledge in G_1 .

Epistemic Agent's Posteriori Syslog Knowledge at T: G_2

G_2 is what the agent knows about a computer's syslog at a particular **T** (time interval). The propositions in G_2 are conceptual graphs as a result of semantic analysis of the message content parsed and tagged using model ***M:***
Log Message:
`session opened for user nobody by (uid=0)`
Tagged Message:
`concept(session),`
`objectAttribute(session,user),`
`attributeValue(status,opened),`
`attributeValue(user,nobody),concept(uid0)`

F generates the semantic meaning of the message:
`opened(session,user(nobody,uid0))`
used in conjunction with the tuple and `Q(Le,T)` as evidence for scenarios in G_1.

WHAT IS A SYSLOG EPISTEMIC AGENT?

An autonomous agent with an epistemic structure E_S that uses RBAC to regulate computer homeostasis. E_S is conceptual graph defined by:

$$E_S = \langle G_1, G_2, V_c, J, F \rangle \text{ where } G_1 \text{ and } G_2 = (V,E)$$

G_1 is **a priori** propositional knowledge of server scenarios in a MTS representation of a POSIX environment.

G_2 is **posteriori** propositional knowledge extracted from syslog entries for some interval $t_2 - t_1$.

J is a poset of abductive justification propositions for one scenarios in G, and ⌟ is defined by:

$$J = \{ p \mid p \subseteq G_1 \wedge \Diamond (G_2 \vdash p) \}$$

where *p* is the most plausible explanation for some conceptual graph g where $g \subseteq G_2$ .V_c is a set of vectors that contains the epistemic agent's level of commitment to J defined by:

$$V_c = \{ c \mid c = \mid\Diamond (G_2 \vdash p) \mid \forall p \in J \}$$