Distributed Fault Correlation Scheme using a Semantic Publish/Subscribe system

Wei Tai, Declan O'Sullivan, John Keeney
Knowledge & Data Engineering Group (KDEG), School of Computer Science and Statistics,
Trinity College Dublin, Dublin, Ireland.
{TaiW | Declan.OSullivan | John.Keeney}@cs.tcd.ie

***Abstract*—Increasingly there is a demand for more scalable fault management schemes to cope with the ever increasing growth and complexity of modern networks. Current distributed fault correlation schemes typically adopt rigid topologies, which require great effort in topology configuration. In this paper, we introduce a distributed fault correlation scheme which addresses this problem by using semantic-based publish/subscribe middleware. This provides loose coupling, robustness and scalability properties for the scheme.**

Keywords; distributed fault correlation, ontologies

I. INTRODUCTION

With the rapid development of network technology and the ever growing demand on networks from both enterprise and network providers, current networks are increasing dramatically both in terms of scalability and complexity. However, traditional fault management approaches, such as OSI management framework [1], typically involve inflexible hierarchical manager/agent topologies and rely upon significant human analysis and intervention, both of which exhibit difficulties as scalability and complexity increases. In addition, traditional centralized event correlation schemes, such as rule-based approaches, codebook approaches [5] and AI approaches [6], are easily flooded by events. However, current distributed correlation schemes, such as the one used in the Madeira project [2], perform distributed correlation by applying centralized correlation schemes at different levels of the managed network or different network domains residing at different geographical locations. By doing so, each Network Element in the managed network needs to be explicitly configured so that all its fault events can be routed to the right correlator, resulting in a tight coupling between the managed network and specific fault management servers through explicit configuration. Alternatively all the fault events can be forwarded to the Fault Management Server using a broadcast approach resulting in a high possibility that event storms will occur.

Our distributed correlation scheme distributes a single correlation task amongst a set of hierarchically arranged correlators. Each correlator takes a fragment of the correlation task and a low level correlator will provide correlation result for higher level correlation. The whole correlation task for the managed network will then be performed hierarchically. Distinguishing from other distributed fault correlation schemes, a semantic publish/subscribe middleware – Knowledge-based Network (KBN) [3] – is used as the underlying event routing mechanism. Both fault events and correlation results will be encoded as KBN notifications and forwarded to appropriate correlators in the KBN who have subscribed to these events and partial correlation results. The usage of KBN loosens the coupling between the managed network and specific correlators. This reduces the possibility of causing an event storm in the correlator network as event correlation can be distributed over different correlators thereby increasing the opportunity for scalability. The circumstances of late delivery and loss of events are also considered in our scheme, and corresponding mechanisms have been devised to address them. Event information and their causal relationships are semantically modeled using ontologies, which are both easy to understand and which provide an opportunity to utilize the reasoning power of ontology reasoners to reduce the complexity of the correlation task. Ontologies play an important role in our work by modeling both the expert knowledge and routing knowledge used in our correlation scheme. Firstly it models expert information and causal relationships of events for the correlation process itself. Secondly, it uses the causal relationships to construct the routing ontology used by the KBN to exchange partially completed correlations in a decentralized manner. By modeling in this way, notification routing based on causal relationships can be achieved, that is an event and all its causes can be routed towards the same correlators. This is achieved in an efficient manner due to the nature by which the underlying KBN is implemented.

Section II provides an overview of the overall architecture. Sections III and IV discuss the semantics of the correlation scheme which is the focus of this paper. Section V briefly describes implementation and evaluation, whilst Section VI draws some conclusions and highlights future work.

II. ARCHITECTURE

The overall architecture of our fault management system comprises three main components: Fault Management Servers (FMS), a Knowledge Based Network (KBN), and Front End (FE) services, as illustrated in Fig. 1.

The Fault Management Server (FMS) is the server on which an event correlator is running. Multiple FMSs can be arranged hierarchically into a network structure named Fault Management Network (FMN) on which the correlation task for the whole managed network will be distributed. Fault correlation is also performed hierarchically. Each low level FMS will pass the result of its correlation fragment to a higher level FMS, and each higher level FMS will then perform correlation over its received lower level events and correlation results. The result of the whole correlation task is available at the FMS on the top of the FMN.

Events are modeled from two aspects: event information and causal relationship. Event information is specific information about this event, such as address where this event was issued (events with the same content but from different network elements are considered as different events and are modeled separately), object identifier and so on. This information could be used for both event correlation and a future fault recovery process. Causal relationships among events can be used in two places: event correlation and distributing partial correlations. For event correlation usage, causal relationships are modeled as correlation rules. This information tells the correlation scheme how a set of events and correlation results (for the sake of simplicity, we use the phrase *correlation elements* to represent "events and correlation results" in the following text) should be correlated into a more meaningful event. For example, the correlation rule {A→C∧D} means "event A can be caused by both event C and D together".

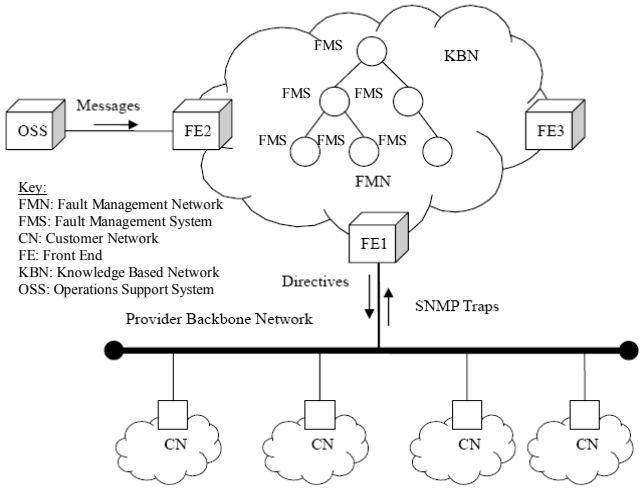

Figure 1: Overall architecture

The Front End (FE) service has been introduced to provide a "two-way" translation between the low level events (SNMP traps) from managed elements and the KBN notifications. It performs the translation by searching the corresponding GUID which uniquely identifies this event within our system, and then a notification will be created by encoding the received trap into the notification's 'payload' along with its 'ID' and semantic 'type' (event or correlation result). This information will then be used by the KBN to route the notification. The mappings between events and KBN messages and vice versa are encoded in a policy-based manner so that they are easy to deploy and replaceable.

III. USING SEMANTICS FOR ROUTING FAULT CORRELATIONS

The KBN is a semantic publish/subscribe infrastructure that extends Siena CBN [4] by the addition of ontology reasoning. It enables the efficient routing of distributed heterogeneous knowledge to, and only to, nodes that have expressed a specific semantic interest in that knowledge. The KBN works as the underlying system and provides a semantic publish/subscribe message transmission mechanism for the inter-connection of FMSs and front end (FE) services. Every functional node running in the fault management system, i.e. FMS and FE, will work as a publisher or a subscriber or both.

As mentioned the casual relationships between event types are modeled as an ontology, whereby the sub/super-class relationship is used to model the "causes"/"caused-by" relationship between event types. This ontology is also used to inform the routing of events and partial fault correlations between correlators according to their semantically-enhanced subscriptions. The control of a correlation fragment for each FMS is then a matter of putting and canceling subscriptions to corresponding events or partial correlations. For example, if a FMS is going to take the correlation fragment which "cause" event A, i.e. subscribe for all correlation element concepts that are a *subclassOf* of semantic concept A. Once subscribed all correlation elements that are the "cause" of A (including A itself) will be forwarded to this FMS while other correlation elements will be automatically filtered by the network.

When this ontology capturing causal relationships between types of events is combined with an ontology to model the topology of the network and its composition of network elements, an ontology of event instances and their causal relationships can be dynamically created. From this the ruleset to drive the correlation activity can also be dynamically built.

IV. DISTRIBUTED CORRELATION SCHEME

The event correlation process on each FMS is divided into adjustable time scales and each time scale is called a *correlation window*. During correlation window, correlation elements will be forwarded to respective FMSs according to the FMS subscriptions. Once a correlation window ends, a new correlation window will start immediately and correlation will be performed over elements that were received in the prior correlation window.

The whole correlation scheme is performed in 4 steps: pre-correlation processing, calculating rules of causal branches, global root cause calculation and estimation, and post-correlation processing. This section will describe this scheme step by step.

Step 1: Pre-correlation processing

At each FMS pre-correlation processing extracts "useful" correlation elements from the message queue for that FMS. This step starts by constructing an empty set, and initializing this set by searching an event(s) (only events) that have the biggest correlation depth – if event A causes B, and B causes C, then in the set {A, B, C} the correlation depth for A is 2 and 1 for B – in the message queue. Then it uses a search algorithm similar to Width First Search to identify all its causes and put them in the set. This search will carry on iteratively until all remaining events can only form sets of cardinality of 1. Since the message queue is a partially ordered set (or poset) with regard to the causal relation "caused_by", the outputs of this step could be more than one subset of the message queue. We call each subset a *sub-task*. For example, considering a FMS with correlation rule set {A→C∧D, B→E∧F} and a message queue {A, B, C, D, E, F}, the outputs of this example should be two sub-correlation tasks: {A, C, D} and {B, E, F}.

As the sub-task search process is running iteratively, the cost could be large. However in our design, the KBN

forwards notifications strictly according to the subscriptions that a FMS has submitted, so it ignores those that are unrelated to this FMS and so this greatly reduces the search task in this step.

Step 2: Calculating Rules of Causal Branches

Correlation Rules only model direct correlation relationships. Therefore, in order to get the *potential* "root causes" of a sub-task, a calculation needs to be performed upon all correlation rules over the causal branch of the sub-task. A *causal branch* of a sub-task is the transitive closure within the correlation fragment of a FMS and it starts from the highest level event of the sub-task. For example, given a sub-task {A, C, D, K} and a set of correlation rules {A→C**Λ**D, C→K**V**G, D→L**V**G}, then the causal branch of the given sub-task is {A, C, D, K, L, G}. This calculation is performed by merging and substituting rules with each other, and it outputs a special correlation rule of which the left side is the event on the highest level and the right side are of the form "disjunction of conjuncts". Each conjunct contains only events that are at lowest level of the causal branch and is regarded as a potential *local root cause* of this sub-task. They are called "local" root cause because they are root causes only within the scope of a sub-task. We call this special rule the *Rule of Causal Branch* (CBR). For example, for a sub-task {A, C, D, K, L}, which requires correlation rules {A→C**Λ**D, C→K**V**G, D→G**V**L}, the CBR will be A→(K**Λ**L)**V**G. For a sub-task {A, D, K, L}, the possible correlation rules and CBR are the same with that of sub-task {A, C, D, K, L}.

Step 3: Global Root Cause Calculation and Estimation

The final goal of event correlation is to determine the *global root cause*, namely, the root cause within the scope of the whole system, for a set of received events. In this step, CBR and correlation results from lower level FMSs are combined to calculate the global root cause for a sub-task. This step can be further divided into three sub-steps: initializing correlation table, calculating global root causes and estimating global root causes.

A. Initializing Correlation Table

All calculations in step 3 will be performed on a table called *correlation table*. The correlation table contains 7 columns. They are: root cause, lost elements, guessed elements, mis-match elements, lost number, guessed number and mis-match number. The root cause is the possible local root cause of the sub-task, which means each conjunct of the CBR for the sub-table will become a row in the table; the next three columns represent the elements that are lost, the elements that need to be guessed to determine the root cause, and elements that do not exist in the root cause (all these are gained by comparing the sub-task list with the potential root cause); the last three columns count the entries in the respective previous columns. Given a sub-task {A, C, D}, its causal branch {A, C, D, E, H} and correlation rule set {A→C**Λ**D, A→E, E→H}, the CBR should be A→(C**Λ**D)**V**H. Then the correlation table should be initialized as Table 1.

root cause	Lost elements	guessed elements	mis-match elements	lost number	guessed number	mis-match number
A→C**Λ**D	None	None	None	0	0	0
A→H	H	H	C, D	1	1	2

Table 1: Initial correlation table

Due to network delay or other reasons, the arrival of elements from the same causal branch could be very sparse, even over two or more adjacent correlation windows. This could put a negative impact on the accuracy of correlation results as well as generate a large number of "garbage elements" (elements that are received but can no longer be used by future correlation). A late delivery handling mechanism is used where the correlator finds that all local root causes listed in the initial correlation table are imperfect, so they have either lost events, or guessed events, or mis-matched events, this round of correlation will be paused to wait for more elements in the next correlation round. This operation is a one-time operation and if all rows in the initial correlation table are still imperfect this mechanism will not be triggered again for this sub-task and the correlation process will carry on.

B. Calculating Global Root Causes

All correlation tables from received correlation results will then be added into the initial correlation table from the previous sub-step. The root cause column will be further calculated by substituting root causes of lower level correlation tables into the root causes of the higher level correlation table. If a newly generated root cause has a form of "disjunction of conjuncts", then that row should also be split into multiple rows with each conjunct a row. The other 6 columns in the new correlation table are recalculated by considering the value of the corresponding columns in the correlation table of lower level correlation results. Continuing our example of the prior sub-step, we assume that the received element C is a correlation result from lower level FMS, which is shown in Table 2 and the final result correlation table is then shown in Table 3.

Root cause	Lost elements	guessed elements	mis-match elements	lost number	guessed number	mis-match number
C→G	G	G	K	1	1	1
C→K	None	None	None	0	0	0

Table 2: Correlation table of correlation result C

Root cause	Lost elements	guessed elements	mis-match elements	lost number	guessed number	mis-match number
A→K**Λ**D	None	None	None	0	0	0
A→H	H	H	C, D	1	1	2
A→G**Λ**D	G	G	K	1	1	1

Table 3: Final result correlation table

Where all received correlation tables have been calculated and all root causes are still not global root causes, the final correlation table obviously cannot be used for estimation as it does not indicate any "real" root causes. Here, an assumption will be made that correlation results have been received causes were lost, e.g. D is not received but L and G are received. The correlator will then create "fake" correlation tables for all lost correlation elements and merge those with the final correlation table. Following the previous example, we assume that D is a low level correlation result but is lost for some reason. Then the correlator finds that all rows in Table 3 do not have global correlation results. It will fake a correlation table like Table 4 and have it merged with Table 3. The new calculated table is the one in Table 5.

Root cause	lost elements	guessed elements	mis-match elements	lost number	guessed number	mis-match number
D→G	D, G	D, G	None	2	2	0
D→L	D, L	D, L	None	2	2	0

Table 4: Fake correlation table of D

C. Estimating Global Root Causes

The sub-step B makes ensures a correlation table containing all possible global root causes is generated. During this sub-step the "real" root cause will be determined amongst the listed global root causes. The determination procedure is based on a belief value calculation: lost number, guessed number and mis-match number are regarded as three belief factors, and different weights (w_1 w_2 w_3) are assigned. The belief value (bv) calculation we used is:

$$bv=1/(lost\_no\times w_1+guess\_no\times w_2+mis\_match\_no\times w_3)$$

The global root cause with the highest belief value will then be regarded as the "real" global root cause for the correlation of this sub-task. From our example, if an element is received, there is a high possibility that something related to that element happened, so we put the highest weight on mis-match number (w_3=3), then the lost number (w_1=2), and the guessed number has the lowest weight (w_2=1). According to this value assignment, we can see that {K, L} is most likely the global root cause of this sub-task.

Root cause	lost elements	guessed elements	mis-match elements	lost number	guessed number	Mis-match number
A→K∧L	D, L	D, L	None	2	2	0
A→G	D, G	D, G	K	2	2	1
A→H	E, H	E, H	C, K	2	2	2

Table 5: final result correlation table with the calculation of D

Step 4: Post-correlation processing

This step chooses whether the correlation table generated by step 3 should be published as a KBN notification or echoed back into the message queue for future correlation. If the highest level element of this sub-task (element A in our previous example) is also the highest level element of the fragment taken by the FMS, then this correlation table should be published to the KBN for the end-point application or higher level FMSs that may have subscribed to this correlation; otherwise, it will be echoed back to this FMS's message queue for use by a subsequent correlation round.

A partial or full correlation result will be created whether it is to be published or echoed back. The correlation result contains a suggested global root cause (as the one estimated in step 3, for human observation) and a correlation table (for higher level FMS consideration). If a correlator can be sure about a correlation result –there exists a row in the output of step 3 that has no lost element, no guessed element and no mis-match element - then only that row will be contained in the correlation table. Otherwise, the whole table will be included in the correlation result.

V. IMPLEMENTATION AND TESTS

This current implementation of the FMS consists of: an Event Correlator developed using Java JDK version 1.5; KBN version 3.0 to route events and messages; and SNMP4J. Currently the SNMP traps are simulated by reading from a file encoded using SNMP4J and then included in a KBN notification that is published. Event information and causal relationships are modeled using OWL-DL.

The average correlation time of this implementation was evaluated using 16 different carefully designed test cases which vary from each other with respect to correlation depth, number of events, number of lost events, and number of correlators. The evaluation result shows that the correlation time decreases dramatically as the number of FMS increases. The result shows that the total correlation time of a correlation task with 13 events, 4 correlation levels requires 325.33 ms when running on one correlator but only 59.33 ms when distributed over 3 correlators.

VI. CONCLUSION AND FUTURE WORK

This paper presented a pure distributed correlation scheme which distributes correlation tasks over a network of Fault Management Servers (FMSs). The distributed correlation scheme presented, which has been implemented and evaluated, allows a single correlation to run in parallel on several correlators to increase the correlation performance.

This scheme used a semantic publish/subscribe middleware (KBN) as the underlying event distribution mechanism, which provided flexibility in topology configuration and reduced the possibility of causing event storms. The advantages of using this KBN approach is that the managed elements and fault management components do not need to be concerned about addressing events, leading to flexibility and scalability.

Additionally, since the correlation behavior is controlled through a subscription mechanism, the correlation behavior of a correlator can be simply changed by altering its subscription to a different part of event correlation graph. This increases the flexibility of the system but also allows the performance of the fault management system to be augmented through the easy addition of new correlators.

In addition, using a subscription-based approach to control the correlation task allows this scheme to be easily extended for robustness, i.e., one FMS can detect the failure of another and replace the failed server by taking over its subscriptions.

This work also demonstrated the benefits of using ontologies to describe the causality relationships between events and partial correlations. The approach also allowed the correlation ruleset to be dynamically created in a network topology-aware manner thereby greatly simplifying the task of defining these correlation rules.

REFERENCES

[1] Yemini, Y., "The OSI network management model," Communications Magazine, IEEE, vol.31, no.5, pp.20-29, May 1993

[2] Arozarena, P., Frints, M., Collins, S., Fallon, L., Zach, M., Serrat, J., and Nielsen, J.. Madeira: A peer-to-peer approach to network management, Wireless World Research Forum, Sep 2006.

[3] Keeney, J., Lewis, D., O'Sullivan, D., "Ontological Semantics for Distributing Contextual Knowledge in Highly Distributed Autonomic Systems", Journal of Network and System Management, March 2007.

[4] A. Carzaniga, D. S. Rosenblum, and A. L.Wolf. Design and evaluation of a wide-area event notification service. ACM Transactions on Computer Systems, 19(3), Aug. 2001.

[5] Yemini, S.A.; Kliger, S.; Mozes, E.; Yemini, Y.; Ohsie, D., "High speed and robust event correlation," Communications Magazine, IEEE , vol.34, no.5, pp.82-90, May 1996.

[6] Huard, J. F., "Probabilistic reasoning for fault management on XUNET". Technical Report, Center for Telecommunications Research, Columbia University, New York, NY, 1994