ATTACK DETECTION AND CLASSIFICATION OF HETEROGENEOUS WIRELESS SENSORS USING CO-CLUSTERING

K.V.RAMANA Ph.D[#1], N.S.V.SRINIVAS[#2]

[#]Computer Science Engineering Department, JNTUK

Kakinada, Andhra Pradesh, 533003, INDIA.

[1]vamsivihar@gmail.com

[2]nsvsrinivas@hotmail.com

Abstract:

In a Wireless Sensor Network a large number of sensors are deployed for the purpose of sensing data and then to bring the data back securely to nearby base stations. The base stations then perform the costly computation on behalf of the sensors to analyze the data sensed by the sensors. Due to resource limitations of the nodes and also due to the vulnerability of physical captures of the nodes, the traditional cryptographic techniques are very complex and should not fit for energy constrained environments.

Data mining techniques can be applied to find the malicious behavior of the nodes of the Sensor Network. By analyzing the traffic patterns one can differentiate the normal behavior from malicious behaviour.Those techniques are used to identify various attacks.

This paper addresses the issue of Attacks using data mining techniques. There exist two types of attacks: (i) External and (ii) Internal. External attacks are those in which an attacker manipulates the communication between pairs of trusted nodes and causes the nodes to de-synchronize. Internal attacks are those in which internal attackers report false clock references to their neighboring nodes proposed an approach to develop a protocol. The protocol not only finds malicious node(s) but also counts them within the group using data mining clustering techniques.

Keywords: Attack Forecasting, Heterogeneous Sensor, Co-Clustering, Attack Graphs, Transitive Closures.

I.INTRODUCTION

Data mining through co-clustering occupies a central role in many fields such as statistics, machine learning, image and video processing, sampling of networking systems, fault diagnosis, performance analysis, and many more. Simultaneously clustering heterogeneous yet correlated sets of objects has attracted much research in recent years, due to the notable impact it has on several application scenarios.

When clustering two heterogeneous sets two dimensional contingency tables can be used to represent their correlations. The task is then to cluster the rows of these tables based on the data reported along the columns, and their columns based on the data reported along the rows. Those two dimensional tables will be treated as adjacency matrix of network graphs. They represent the co-relation among various groups of nodes.

The utility of organizing combinations of network attacks as graphs is well established. Traditionally, such attack graphs have been formed manually by security red teams (penetration testers). But significant progress has been made recently in generating attack graphs automatically, based on models of network security conditions and attacker exploits, created from network scans, vulnerability databases, etc. By representing dependencies among attacker exploits rather than explicitly enumerating attack states, exponential graph complexity can be avoided.

In the current state of practice, it is thus possible to efficiently compute attack graphs for realistic networks. But the resulting graphs can still pose serious challenges for human comprehension. This is compounded by the fact that attack graphs are usually communicated by literal drawings of graph vertices and

edges. While graph drawing has been studied extensively, the problem is ill posed in the sense that many possibilities exist for what constitutes a good graph drawing. Also, finding optimal placement of graph vertices according to many of the desired criteria is NP-complete.

II.PROPOSED APPROACH

A. ARCHITECTURE

As described in [3] approach is to capture the network configuration, from which predict attack paths through the network, and use the predicted paths for sensor placement, alarm prioritization, and attack response. As shown in Fig 1, scan the network to discover hosts, their operating system, application programs, and vulnerable network services. Proposed approach can also capture network connectivity, including the effects of devices such as firewalls and router access control lists. With the resulting network model, a database of modeled attacker exploits, and a specification of threat origin and critical network assets, it computes an attack graph, as described in. This graph, computed in worst-case quadratic time, comprises all known attacks through the network.

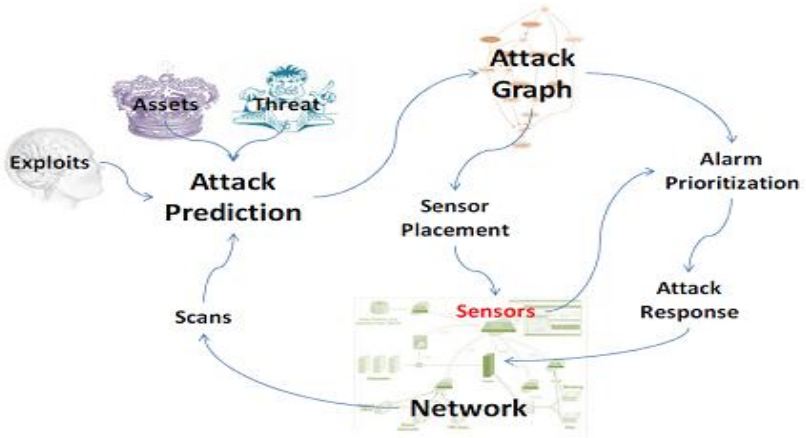

Fig.1: System Model

Proposed approach applies information-theoretic co-clustering to the attack graph matrix with join probability distribution. This clustering rearranges rows and columns of the adjacency matrix to form homogeneous groups. In this way, overlapped area of patterns of the matrix represents the co-relation, and groups (attack graph subsets) can be considered as single units. This clustering technique is fully automatic, is free of parameters, and scales linearly with graph size.

Fig. 2 represents all the possible paths those cover all the vulnerable connections in the attack graph. Using this approach algorithm will take polynomial time to find optimal solution.

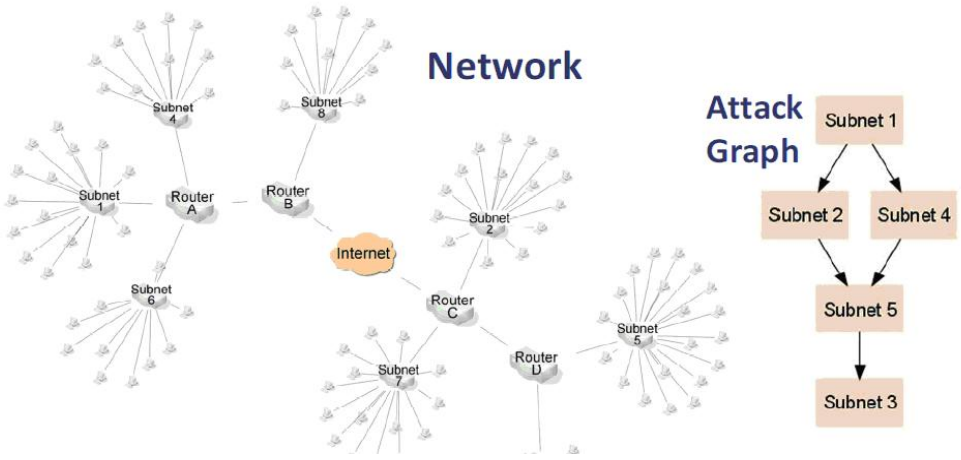

Fig.2 : Example Network and a High-Level Summary of Its Attack Graph

The general approach is to find clustering attack graph with adjacency matrices, thus finding transitive closures, which provides a framework for correlating, predicting, and hypothesizing about network attacks. The approach applies to general attack graphs, regardless of what the particular graph vertices and edges represent. For example, such attack graphs could have been formed from models of network vulnerability, or from causal relationships among intrusion detection events. Attack graph vertices could also represent aggregated sub graphs, such as aggregation by machines and exploits between them. Overall, the techniques describe here have linear time complexity in the size of the attack graph, for scalability to larger network.

As a data structure, an alternative to adjacency matrices are adjacency lists. For each vertex in the graph, the adjacency list keeps all other vertices to which it has an edge. Thus, adjacency lists use no space to record edges that are not present. There are tradeoffs (in both space and time) between adjacency matrices and lists, depending on graph sparseness and the particular operations required.

Formally, for n vertices in the attack graph, the adjacency matrix A is an n × n matrix where element $a_{i,j}$ of A indicates the presence of an edge from vertex i to vertex j. In attack graphs, it is possible that there are multiple edges between a pair of vertices, in such cases, one can either record the actual number of edges, or simply record the presence (0, 1) of at least one edge. The adjacency matrix records only the presence of an edge, and not its semantics, which can be considered in follow-on analysis

There is a need to apply any matrix co-clustering algorithm that is designed to form homogeneous rectangular blocks of matrix elements. Here, homogeneity means that within a block, there is a similar pattern of attack graph edges. This clustering algorithm is a non-parametric and scales linearly with problem size.

By using the co-clustering algorithm in [2] for attack graph adjacency matrices proposed work can formulate the problem of attack prediction and isolate the sensor nodes which are compromised by the adversaries. Overview of the selected algorithm as follows in Fig 3

Input: Domains D_X, D_{Y1},...,D_{YN}, real nums β_1, ..., β_N, cluster sets $\widehat{D}_X$, $\widehat{D}_{Y1}$,...,$\widehat{D}_{YN}$, and joint distributions $p_1(X,Y^1),...,p_N(X,Y^N)$;
Output: A co-clustering for Y^1,...,Y^N w.r.t. X;

Define an arbitrary co-clustering $\langle C_X^0, C_{Y1}^0, ..., C_{YN}^0 \rangle$;
set $t=0$, $\Delta I_i^{(0)} = +\infty$, and compute $q_i^{(0)}$, for $i = 1..N$;
repeat
  for each Y^i and $y \in D_{Y^i}$ **do**
    $C_{Y^i}^{(t+1)}(y) = \arg\min_{\hat{y} \in \widehat{D}_{Y^i}} \mathcal{D}(p_i(X|y)||q_i^{(t)}(X|\hat{y}))$;
  Compute $q_i^{(t+1)}$, for $i = 1..N$;
  for each $x \in D_X$ **do**
    $C_X^{(t+2)}(x) = \arg\min_{\hat{x} \in \widehat{D}_X} \sum_{i=1}^N \beta_i \cdot \mathcal{D}(p_i(Y^i|x)||q_i^{(t+1)}(Y^i|\hat{x}))$;
  Compute $q_i^{(t+2)}$, for $i = 1..N$;
  let $\Delta I_i^{(t+2)} = \mathcal{D}(p_i(X,Y^i)||q_i^{(t+2)}(X,Y^i)), \forall Y^i$;
  set $t = t+2$;
until $\sum_{i=1}^N \beta_i \cdot \Delta I_i^{(t)} = \sum_{i=1}^N \beta_i \cdot \Delta I_i^{(t-2)}$;
return $\langle C_X^{(t)}, C_{Y1}^{(t-1)}, ..., C_{YN}^{(t-1)} \rangle$;

Fig. 3**:** LC-HOCC Algorithm

B. MULTI HOP CONNECTIVITY

The adjacency matrix shows the presence of each edge in a network attack graph. Taken directly, the adjacency matrix shows every possible single-step attack. In other words, the adjacency matrix shows attacker reachability within one attack step. As we describe later, one can navigate the adjacency matrix by iteratively matching rows and columns to follow multiple attack steps. But as an alternative, raise the adjacency matrix to higher powers, which shows multi-step attacker reachability at a glance. A fundamental property of attack graphs is how well connected the various graph vertices (exploits, machines are. Knowing the numbers and depths of attacks helps us understand large-scale tendencies across the network

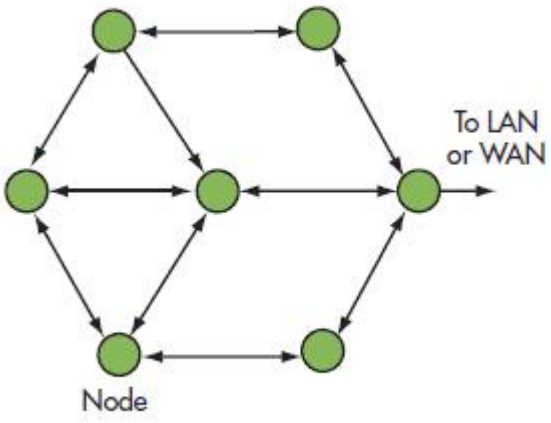

Fig. 4: The Nodes Communicate with One Another and Offer a Multi-Hop Capability.

C. ATTACK FORECASTING

One can place detected intrusions within the context of vulnerability-based attack graphs. By associating intrusion alarms with reach ability graph, and also predict the origin and impact of attacks. That is, once place intrusion alarm on one of the vulnerability-based reach ability graphs; navigate the graph to do attack prediction. The idea is to project to the main diagonal of the graph, in which row and column indices are equal. Vertical projection (along a column) leads to attack step(s) in the forward direction. That is, when one project along a column to the main diagonal, the resulting row gives the possible steps forward in the attack.

III.CONCLUSION

This paper analyzes how co-clustered adjacency matrices reveal the underlying regularities in network attack restrictions on the form of the attack graph. It therefore applies to attack graphs based on network vulnerabilities, detected intrusions, or combinations thereof, and well as attack graphs with aggregated vertices, e.g., aggregated by network machine. The information-theoretic co-clustering algorithm for multiple heterogeneous environments by dividing the matrix into rows and columns of the adjacency matrix so that rectangular blocks of similarly-connected attack graph elements emerge. This clustering algorithm is fully automatic, parameter-free, and scales linearly with problem size.

REFERENCES

[1] I.F.Akyildiz, W.Su,Y. Sankarasubramaniam, and E. Cayirci, “Wireless Sensor Networks : A survey,” Computer Networks, vol. 38, no. 4, pp.393–422, 2002

[2] Gianluigi Greco, Antonella Guzzo, Member, IEEE, and Luigi Pontieri "Coclustering Multiple Heterogeneous Domains:Linear Combinations and Agreements", VOL. 22, NO. 12 pp.1649-1663, DECEMBER 2010

[3] S. Noel, E. Robertson, S. Jajodia, “Correlating Intrusion Events and Building Attack Scenarios through Attack Graph Distances,” in Proceedings of the 20th Annual Computer Security Applications Conference, Tucson, Arizona, December 2004.

[4] S. Noel, S. Jajodia, “Managing Attack Graph Complexity through Visual Hierarchical Aggregation,” in Proceedings of the ACM CCS Workshop on Visualization and Data Mining for Computer Security, Fairfax, VA, October 2004.

[5] P. Ning, D. Xu, C. Healey, R. St. Amant, “Building Attack Scenarios through Integration of Complementary Alert Correlation Methods,” in Proceedings of the 11th Annual Network and Distributed System Security Symposium, February 2004.

[6] S. Noel, S. Jajodia, B. O’Berry, M. Jacobs, “Efficient Minimum-Cost Network Hardening via Exploit Dependency Graphs,” Proceedings of the 19th Annual Computer Security Applications Conference, Las Vegas, Nevada, December 2003

[7] P. Berkhin and J.D. Becher, “Learning Simple Relations: Theory and Applications,” Proc. SIAM Int’l Conf. Data Mining (SDM ’02),pp. 420-436, 2002.

[8] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D.S. Modha, “A Generalized Maximum Entropy Approach to Bergman Co-Clustering and Matrix Approximation,” Proc. Int’l Conf. Knowledge Discovery and Data Mining (KDD ’04), pp. 509-514, 2004.

[9] L. Zhao and M.J. Zaki, “Tricluster: An Effective Algorithm for Mining Coherent Clusters in 3D Microarray Data,” Proc. ACM SIGMOD, pp. 694-705, 2005.

[10] B. Gao, T.-Y. Liu, X. Zheng, Q.-S. Cheng and W.-Y. Ma, “Consistent Bipartite Graph Co-Partitioning for Star-Structured High-Order Heterogeneous Data Co-Clustering,” Proc. Int’l Conf. Knowledge Discovery and Data Mining (KDD ’05), pp. 41-50, 2005