Intelligent Network Management Using Graph Differential Anomaly Visualization

Qi Liao
Department of Computer Science
Central Michigan University, USA
Email: qi.liao@cmich.edu

Aaron Striegel
Department of Computer Science and Engineering
University of Notre Dame, USA
Email: striegel@nd.edu

***Abstract*—Managing large-scale networks involving users and applications is challenging due to the complexity and dynamic nature of the heterogeneous graphs. How to quickly identify the meaningful changes and hidden anomalous activities in the spatiotemporally dynamic network graphs is essential in many aspects of network management, such as security, performance and troubleshooting. In this paper, we explore the viability and efficacy of a novel graph differential anomaly visualization (DAV) model in the area of network management. Our approach combines algorithmic graph analysis methods and visualization technologies by taking advantages from both computer and human intelligence. We focus on DAV at various levels, i.e., nodes, links and communities. Specifically, a novel community-based DAV scheme is proposed that can help understand the managed networks with a right balance of granularity and complexity. More importantly, the community-based DAV algorithm is less susceptible to network dynamics and high churn. The developed visual analytic tool can not only detect but more importantly find the root causes of anomalies in a time efficient manner.**

I. Introduction

Managing large-scale enterprise networks is hard due to the complexity and dynamics of the network activities. Particularly, the *context* of connections, i.e., the users (*who*) and applications (*what*) that are responsible, is more difficult to manage than simply *where* (address/port). With the trend of moving computation and data into the cloud, the traditional sense of physical location of hosts becomes less precise in describing what is going on in the networks. Instead, users, applications and data are receiving increasing attention from the perspective of network management [1]–[3].

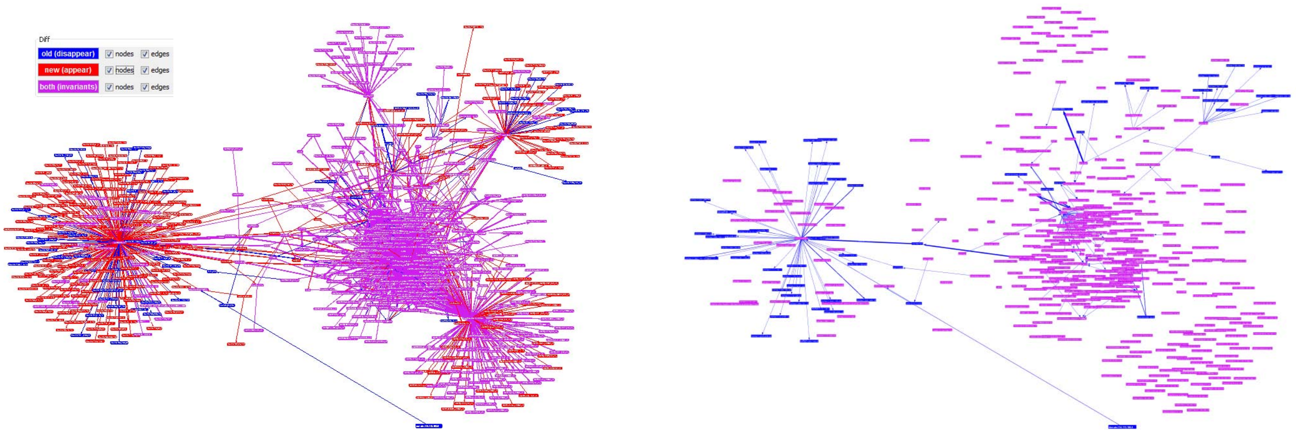
Effective network management requires network operators and managers to understand not only what is happening on the network but what are the abnormal changes as well. However, the challenge brought by the user and application activities makes the network more complicated and dynamic as network graphs are constantly changing. Being able to quickly understand and identify what the important changes are can have a significant impact on many aspects of network management. For example, for security management, anomalous user or application behaviors could indicate intrusions and attacks. In performance management, anomalous changes of routing paths or traffic patterns may indicate degraded services and utilization. For fault management, the difference between two snapshot network graphs could help troubleshooting connectivity problems, e.g., application *A* should have (but not) contacted a specific server *S* at time *t*.

978-1-4673-0269-2/12/$31.00 © 2012 IEEE

While techniques in data mining and machine learning can help to some degree (despite high false positives), these approaches alone are less effective in network management [4], [5]. Visualization, on the other hand, can be useful for network managers to quickly overview the managed network (situation awareness) [6], but is only useful if the investigator knows exactly what to look for.

In this paper, we explore the feasibility and efficacy of a smarter network management scheme by combining both algorithmic data analysis methods and interactive visual data exploration. Specifically, we developed a visual analytic tool based on a technique named graph *differential anomaly visualization* (DAV). In addition to magnitude-based anomalies (e.g., massive port scans and DoS attacks), which can be relatively easily picked up by traditional intrusion detection system (IDS), we focus on more general term of anomalies (e.g., users and applications may not necessarily incur lots of traffic but slightly change their connection behaviors perhaps due to malicious intention or misconfiguration). Essentially, we are looking at a harder problem that given only snapshot of time-series network graphs without any priori knowledge of good or bad, can we detect abnormal changes and the underlying causes? The key challenge is therefore how to effectively visualize the *dynamics* and *similarity* (or conversely *difference*) among the heterogeneous network graphs consisting of hosts, users, and applications. The ability to extract the meaningful changes from otherwise dynamic and noisy network data and present them in a visually appealing manner that can provide insight to network management is non-trivial.

The contribution of the paper consists of a visual analysis framework that utilizes the *differential anomaly visualization* (DAV), which is based on the evolution of network graphs ranging from the details of *nodes/edges* to the abstraction of *communities*. While analyzing the overall network properties might be too coarse to be useful for network management, a novel community-based DAV scheme is proposed that can help understand the managed networks with a right balance of granularity and complexity. *More importantly, the community-based DAV algorithm is more tolerant to the high dynamics of network by treating communities rather than individual nodes or edges*. In addition, a novel link anomaly detection and


Diff
old (disappear) ☑ nodes ☑ edges
new (appear) ☑ nodes ☑ edges
both (invariants) ☑ nodes ☑ edges

(a) Complete view: *Blue*: appeared only in the first graph; *Red*: appeared only in the second graph; *Purple* (*red+blue*): appeared in both graphs.

(b) Filtered view: shows nodes/edges (blue) that disappeared from time t to t' relative to more stable nodes (purple).

Fig. 1: Differential visualization of *HUA* graphs at the node/edge level.

visualization algorithm is also proposed that has a potential impact on research community of network operation and management. While security management is the focus of this study, the proposed anomaly analytic methodologies has potential applications to other important aspects of network operations and management such as performance management and fault management.

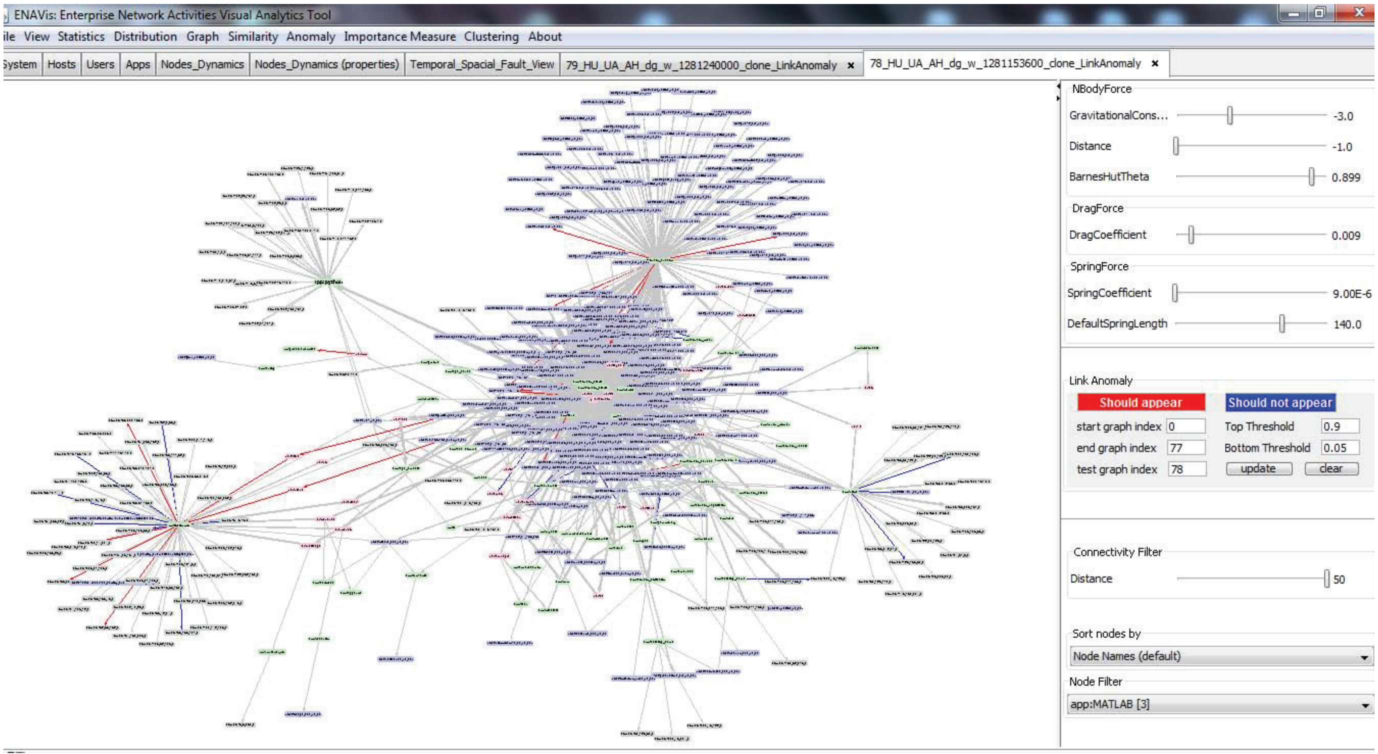
II. Related Work

In areas of network security management, intrusion and anomaly detection [7] can be roughly categorized as signature-based, statistical/mining-based, and visual-based. While signature-based schemes have the advantage of low false positives, signatures require well defined patterns in advance making the detection of zero-day exploits impossible as well as less effective for encrypted activities or self-modifying worms. Machine learning based anomaly detection is promising since pre-defined signatures are not required. However, data mining and machine learning technologies alone are not enough because there is lack of attack-free clean training data [4] and there are certain patterns or knowledge that can be missed by traditional automatic data mining methods [5]. Furthermore, it is hard to bridge the gap between the data mining results and their operational interpretation (e.g., if an IDS gives alarms, one needs to know *why* and what the alarms mean?). How to effectively analyze the *causes* of those anomalies is the key for successful and *time efficient* troubleshooting and diagnosing of network problems.

Visualization-based anomaly detections require human interaction and domain knowledge. While visualization can be a promising approach to find and understand the root cause of various network abnormalities, out of the few existing visualization tools [8]–[12], most rely on either packet-level or flow-level information. These visualizations often fall short in enterprise settings where users and applications are more important from a security policy perspective than the particular host IP and/or port [1], [2]. To understand and troubleshoot dynamic, large-scale enterprise networks that involve users and applications with unknown anomalies, our work aims at bringing both computer and human intelligence that can effectively analyze the *causes* for hard-to-detect anomalies which are essential in network security, performance and fault management.

Complex systems can often be represented as network graphs. There have been research communities focusing on graph mining [13], [14], in which community detection or clustering [15], [16] algorithms and link prediction algorithms [17], [18] can be useful in understanding the networks. However, most works in link prediction are only interested in predicting whether a pair of nodes that are previously not connected will ever be connected in the future. Therefore, link predictions do not focus on the task of link anomalies [19], [20] and do not address more *dynamic* anomaly issues, e.g., whether a previously connected link will become disconnected, or whether and when links will have “on/off” behavior. The visual analysis framework in this work allows easy integration of any link anomaly detection algorithms, which is an interesting yet challenging topic that can have great potential in network management.

This work extends our earlier work *ENAVis* [3], [21], which primarily focused on the visualization and exploration aspect of network management data involving hosts, users and applications. This paper focuses on dynamics, particularly high churn (user/app nodes come and go). Measuring the differences at node and edge level will produce lots of fluctuation. We argue the view from the community level is less susceptible to network dynamics. In particular, this paper presents a novel community-evolution-based graph differential anomaly visualization (DAV), which is more tolerant to the high dynamics typically involved in users and applications. In addition, the proposed link anomaly visualization is another interesting topic that may, in our hope, invoke discussions with other researchers in network management area.


ENAVis: Enterprise Network Activities Visual Analytics Tool
ile View Statistics Distribution Graph Similarity Anomaly Importance Measure Clustering About
System
Hosts
Users
Apps
Nodes_Dynamics
Nodes_Dynamics (properties)
Temporal_Spacial_Fault_View
79_HU_UA_AH_dg_w_1281240000_clone_LinkAnomaly
78_HU_UA_AH_dg_w_1281153600_clone_LinkAnomaly
Should appear
Should not appear
update
clear

NBodyForce

GravitationalCons... -3.0

Distance -1.0

BarnesHutTheta 0.899

DragForce

DragCoefficient 0.009

SpringForce

SpringCoefficient 9.00E-6

DefaultSpringLength 140.0

Link Anomaly

start graph index 0

end graph index 77

test graph index 78

Top Threshold 0.9

Bottom Threshold 0.05

Connectivity Filter

Distance 50

Sort nodes by

Node Names (default)

Node Filter

app:MATLAB [3]

Fig. 2: Screenshot of link anomaly visualization. Filtering options on the right panel allow users to adjust anomaly threshold and range of graphs under investigation.

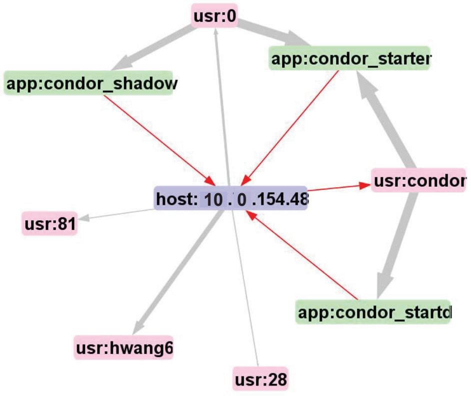


Fig. 3: Example of link anomaly visualization: three *condor*-related applications and the condor users should have run on a host 10.0.154.48 but did not.

III. Differential Anomaly Visualization on Nodes and Edges

Understanding the spatio-temporal differences (or conversely similarities) among network snapshot graphs is usually the first important step to detect and analyze network abnormalities. In this section, we focus on the differential anomaly visualization (DAV) in detail of *nodes* and *edges* and link anomaly visualization. In Section IV, we will study DAV in terms of evolution of graph community structures.

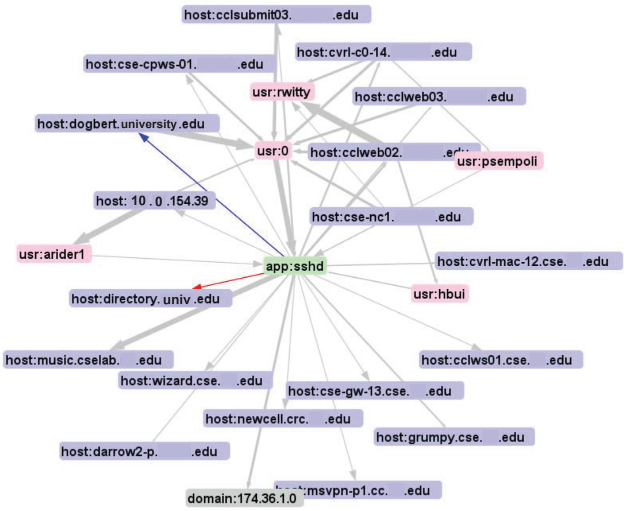


Fig. 4: Example of link anomaly visualization of a *HUA* graph showing both two types of anomaly links.

A. Data collection and graph construction

Based on the observation that only the end hosts can offer maximum visibility of the user and application activities, data collection agents [3] have been deployed at the university campus that includes a mixture of faculty/students' desktop computers, machines from computer labs and scientific computing pools (e.g., *condor* [22]). The data gathering component of the Linux version of the agent utilizes commonly available system tools such as `netstat` and `ps` in order to take advantage of administrator familiarity, development easiness and robustness. The several hundred nodes we monitor are maintained by a simple *up2date* script for install and health checking. The primary purpose of the agents is to collect the *local context* information, i.e., which users running what applications associated with each network connection. The context graph data in GraphML [23] format is then fed into the visualization tool.

B. DAV on nodes and edges

To analyze the dynamic network graphs, one can focus on the overall graph properties. While this may be a useful starting point, many times analyzing graph property changes is too coarse to be of any particular usefulness for network administrators, who usually want to find out what exactly goes wrong. For example, two networks can have exactly the same degree distribution but are totally different. On the other hand, one can analyze the exact graph changes in terms of individual nodes and edges, which is similar to graph edit distance [24]. Our tool incorporates information visualization techniques that allow investigators to explore the data in intuitive graph views with the changes among snapshots highlighted in appropriate color codings.

Figure 1a shows DAV on *HUA* snapshot graphs. The tool allows zoom-in/out function on the graphs for both overview and details. Options are also available to allow filtering. For example, Figure 1b shows a combination of old nodes/edges (blue) with invariant nodes only (purple). It clearly shows all nodes/links that disappear from the previous snapshot graph. This visual analysis can be very helpful for human investigators to gain a quick overview on temporal changes. For example, comparing a healthy network at time t with a faulty or compromised network at time t' can reveal insights on possible reasons causing the problem.

C. Link Anomaly Visualization

Figure 2 shows a screenshot of link anomaly visualization. Different threshold can be selected on the right filtering-option panel. If there are links that should appear but did not appear on a specific snapshot graph, *red* colors are used to denote the anomalous edges (Type-I). On the other hand, if there are links that should *not* appear but actually appeared at a specific time, then blue colors are used to denote the anomalous links (Type-II).

1) Link Anomaly Detection: A proof-of-concept algorithm for detecting the above Type-I and Type-II link anomalies is defined as follows:

$$P(L_i) = \frac{\sum_{t=1}^{N} w(t) \cdot d}{\sum_{t=1}^{N} w(t)}, \; d_{t,i} \in \{0, 1\} \tag{1}$$

$$w(t) = e^{-\lambda\left(1-\frac{t}{N}\right)} \tag{2}$$

The appearance probability functions can be either weighted or unweighted. The weighted form (Equation 1) takes a non-linear time weighting function $w(t)$ (Equation 2), i.e., the appearance of links at later snapshot graphs (or in other words closer to the time of investigation interest) should have higher weights over the earlier graphs. Both Equations 1 and 2 are normalized between 0 and 1, where $P(L_i)$ represents the probability of *i*th link; N denotes the number of snapshot graphs; and $d_{t,i}$ takes a binary form to denote whether *i*th link appears or not at time t.

2) Examples: Figure 3 shows one user (*condor*) and three *condor*-related applications (*condor_shadow*, *condor_starter* and *condor_startd*) that should have run (above 90% probability) on the host *10.0.154.49* but did not appear in one snapshot graph, indicating potential problems with *condor* dispatcher and services. Figure 4 demonstrates both Type-I (red) and Type-II (blue) anonymous links. Users interact with the application *sshd*, which contacts a suite of hosts with different probabilities. In this specific example, *dogbert.university.edu* has a very low probability to appear but actually appeared on that day while *directory.univ.edu* has a high probability to appear but did not. These link anomaly visualization examples have security and fault implications. While the proposed link anomaly detection algorithm (Equation 1) is relatively simple for illustration purpose, the link anomaly visual analytic framework can be extended to future sophisticated link anomaly detection algorithms.

IV. Differential anomaly visualization via community evolution

While the graph differential anomaly visualization discussed in Section III gives the maximum details in terms of which hosts, users and applications, we are further interested in analyzing spatio-temporal anomalies by taking a right balance of granularity and complexity. As stated earlier, the measurement of overall graph properties such as degree distributions is too coarse to be useful. On the other hand, analyzing every single change happening on the node/edge levels can be of too much details and obfuscating, thus less effective in face of larger networks with higher dynamics and churn rates.

We develop a visual analytics function based on community membership changes. One can view this approach as an *intermediate* similarity metric between the levels of graph properties (coarse) and nodes/edges (fine). The immediate advantage of comparing networks at the community level is the attenuation of noise from individual node/edge changes. One key challenge behind anomaly analysis is to ask what changes are normal while other changes are abnormal, and what are the reasons behind these anomalies. In the scheme of community-based graph DAV, no matter how dynamic the nodes are (come and go), if nodes consistently belong to the same community (or consistently belong to different communities), it is considered normal change; otherwise, it is considered abnormal change.

A. Concept Illustration

Intuitively, if a user suddenly uses a different set of applications, appears on a different set of hosts, or contacts many different target machines causing his membership change with

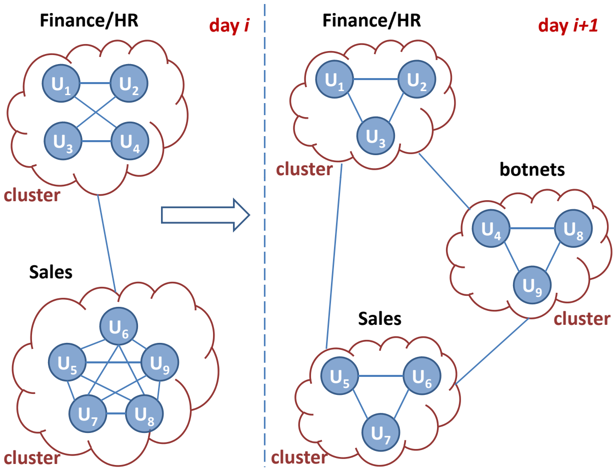


Fig. 5: Illustration of graph differential anomaly visualization through community membership changes.

respect to other users, then the behavior of that user is at least suspicious and needs the administrator's attention for further investigation.

Figure 5 illustrates the concept of community-based graph differential anomaly visualization: users may switch their memberships in a temporal manner. This example uses a *user similarity graph*, in which edges suggest neighboring users share at least one destination host. The distance between two snapshot graphs increases as some user-nodes change their cluster memberships from Finance/HR and Sales departments to an unknown community over time. Further investigation reveals that those users (U_4, U_8 and U_9) all contacted $cc3.irc.ru$, which is one of the command and control (C&C) channels controlled by certain botmasters. Therefore, these users could have been compromised and become a part of a larger botnet.

Algorithm 1 Compute normalized differences between two sets of communities

```
Require: $C_1$, $C_2$
Ensure: distance between $C_1$ and $C_2$
  N := $C_1 \cup C_2$
  for each pair of nodes $(n_i, n_j) \in N$ do
      if $n_i$ & $n_j$ belong to the same cluster $\in C_1$ then
          if $n_i$ & $n_j$ belong to the same cluster $\in C_2$ then
              $SS := SS + 1$
          else
              $SD := SD + 1$
          end if
      else
          if $n_i$ & $n_j$ belong to the same cluster $\in C_2$ then
              $DS := DS + 1$
          else
              $DD := DD + 1$
          end if
      end if
  end for
  return $1 - (SS + DD)/(SS + SD + DD + DS)$
```

B. Algorithm

We use a simple yet effective algorithm to measure the changes of communities. Given any two sets of communities or clusters C_1 and C_2, which do not have to contain exactly the same number of communities, the distance between communities is based on an idea derived from the Rand Index [25], i.e., by taking the ratio of how many nodes consistently belonging (or not belonging) to the same community over those belonging to the same community in C_1 but end up in different community in C_2 or vice verse. The higher the ratio, the smaller the distance, as shown in Algorithm 1.

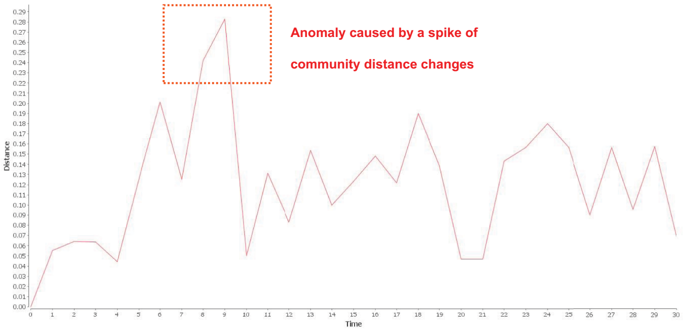


Fig. 6: Community-based graph differential anomaly visualization from Algorithm 1. Anomalies are suggested by the spikes of graph distances in terms of normalized percentage of community changes.

Once we compute a distance matrix over all pairs of communities, a multidimensional scaling (MDS) [26] view can be mapped in an efficient way that allows the human investigator to spot easily any changes (or anomalies) over the entire data range. With the help from the intelligence provided by the visual analytic tool, the administrator's domain knowledge can then play an important role when drilling down the suggested anomaly to the root cause, i.e., what actually causes these changes, by interacting with the data through user-friendly clicks and queries.

C. Case Study

Suppose an administrator wants to explore his log data and to see if there are any suspicious user behaviors that possibly violate acceptable use policy (AUP). He opens the graph differential anomaly visualization tool, and sets the granularity of time window as one day and thirty daily network activity snapshot graphs are generated automatically by the visualization tool. Since the administrator only wants to see user behaviors, he chooses to generate *user similarity graphs*. For example, two user nodes are connected only if they share at least one common application. With a click of menu option, various graph community detection algorithms can be applied. In this case, the *Walktrap* [16] algorithm is selected to compute the optimal communities. The *Walktrap* is especially appealing because the administrator does not need to specify the exact number of communities in advance.

Figure 6 shows the community distance changes (computed by Algorithm 1) over the one month's period. Distances are in proportion to community membership changes between consecutive snapshot graphs. Therefore, the spikes in the distance indicate potential anomalies. In this example, graphs

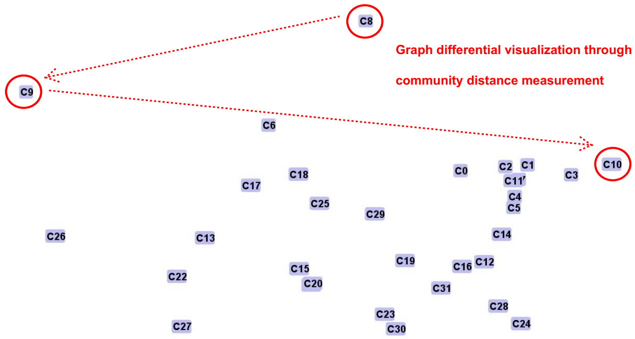
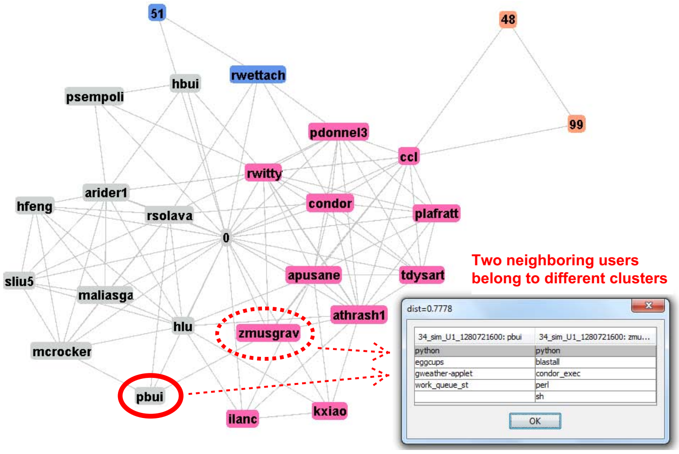


Fig. 7: An alternative MDS overview reflects the evolution of community memberships of snapshot graphs. C_i represents the set of communities of the ith graph. Nodes that are further away indicate anomalous user behaviors.

from day 8 to day 9 and from day 9 to day 10 have the highest percentages of community membership changes. As a useful compliment, a 2D MDS view (Figure 7) is generated in another window to visually analyze the overall community evolution. In this view, each node represents one daily snapshot graph containing a set of communities. The distances between these nodes reflect the similarity levels among those daily snapshot graphs based on the metric of community membership changes. The larger the distance and further away from the rest of graphs, the more anomalous the network graphs are. Figure 7 intuitively matches Figure 6 with significantly larger distances among C_8, C_9 and C_{10}. Interestingly, C_{10}, despite its large distance from the immediately previous graph, returns to a state closer to many of previous graphs, i.e., $C_0 - C_5$.

A natural question following the above observation is who is responsible for those changes. The interactive exploration capability of the tool allows the investigator to visually explore the two graph communities (Figure 8 and Figure 9) that correspond to the time of change (day 8 to day 9). In the previous snapshot graph (Figure 8), one enterprise user *pbui* belongs to the community largely formed by graduate students who run a similar set of Linux desktop applications that make network connections. The user *pbui* shares one application (*python*) with one of his neighboring nodes (*zmusgrav*), who nevertheless belongs to another community formed largely by *condor* users. Interestingly, in the following snapshot graph (Figure 9), the same user *pbui* changes his community membership to the *condor* community. The querying mechanism is enabled via multiple node selections by holding down `Ctrl` key and right-clicking nodes for comparing overlapping attributes (shared application in this case). The query suggests that the new application, i.e., *condor_shadow* used by the user *pbui*, causes the membership changes.

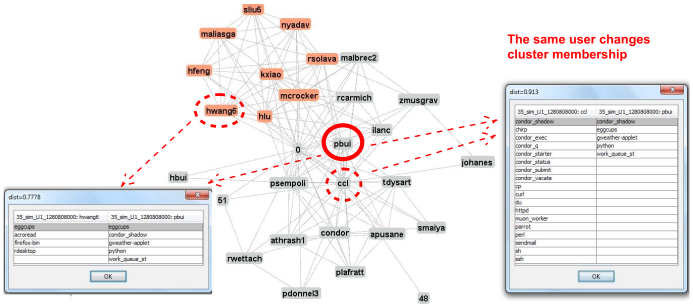
While the data in the above example are directly from users consisting of mostly students and faculty and may not contain malicious attacks, the methodology of the proposed algorithms and visualization framework allows the detection of potential malicious user behaviors possible. Graph differential anomaly visualization based on community membership changes can serve as a promising alternative in analyzing hard-to-detect anomalous network activities that worth further investigation.



dist=0.7778

34_sim_U1_1280721600: pbui	34_sim_U1_1280721600: zmu...
python	python
eggcups	blastall
gweather-applet	condor_exec
work_queue_st	perl
	sh

Fig. 8: Community visualization at time 8. Specifically, enterprise user *pbui* belongs to the graduate student community (gray) while one of his neighbors *zmusgrav* belongs to condor community (pink). Popup window returns query result showing the two users share one application *python*.



dist=0.7778

35_sim_U1_1280808000: hwang6	35_sim_U1_1280808000: pbui
eggcups	eggcups
acroread	condor_shadow
firefox-bin	gweather-applet
rdesktop	python
	work_queue_st

dist=0.913

35_sim_U1_1280808000: ccl	35_sim_U1_1280808000: pbui
condor_shadow	condor_shadow
chirp	eggcups
condor_exec	gweather-applet
condor_q	python
condor_starter	work_queue_st
condor_status	
condor_submit	
condor_vacate	
cp	
curl	
du	
httpd	
muon_worker	
parrot	
perl	
sendmail	
sh	
ssh	

Fig. 9: Community visualization at time 9. The enterprise user *pbui* migrates from the graduate student community to the condor community (gray) by using a new application *condor_shadow*.

V. Conclusion

Managing large-scale complex networks is challenging due to the dynamics of increasing involvement of users and applications in the enterprise network activities making traditional monitoring and analysis mechanism less effective. In this study, we developed a novel visual analytic tool that combines both human and computer intelligence for smarter network operations and management. In particular, the graph differential anomaly visualization framework, which is based on both individual nodes/links and the evolution of community structures, can detect and find the root causes of anomalies in dynamic graphs. The proposed graph anomaly visualization algorithm has direction applications in network security management with potential usefulness in other network operation and management areas.

Acknowledgment

The authors would like to thank Dr. Nitesh Chawla and Ryan Lichtenwalter at University of Notre Dame for valuable comments on graph mining and link prediction, and Dr. Lei Shi at IBM Research China for helpful suggestion on visualization design.

References

[1] P. Hertzog, "Visualizations to improve reactivity towards security incidents inside corporate networks," in *Proceedings of the 3rd International Workshop on Visualization for Computer Security (VizSec '06)*, Alexandria, Virginia, November 3 2006, pp. 95–102.

[2] D. Lalanne, E. Bertini, P. Hertzog, and P. Bados, "Visual analysis of corporate network intelligence: Abstracting and reasoning on yesterdays for acting today," in *Proceedings of the 4th International Workshop on Visualization for Computer Security (VizSec'07)*, Sacramento, CA, October 29 2007, pp. 115–130.

[3] Q. Liao, A. Blaich, A. Striegel, and D. Thain, "ENAVis: Enterprise network activities visualization," in *Proceedings of the USENIX 22nd Large Installation System Administration Conference (LISA '08)*, San Diego, CA, November 9-14 2008, pp. 59–74.

[4] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *IEEE Symposium on Security and Privacy*, Oakland, CA, May 16-19 2010, pp. 305–316.

[5] S. T. Teoh, K.-L. Ma, S. F. Wu, and T. Jankun-Kelly, "Detecting flaws and intruders with visual data analysis," *IEEE Computer Graphics and Applications*, vol. 24, no. 5, pp. 27–35, September/October 2004.

[6] R. Marty, *Applied Security Visualization*. Addison Wesley Professional, 2009.

[7] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, article 15, pp. 1–58, July 2009.

[8] J. Goodall, W. Lutters, P. Rheingans, and A. Komlodi, "Focusing on context in network traffic analysis," *IEEE Computer Graphics and Applications*, vol. 26, no. 2, pp. 72–80, March/April 2006.

[9] G. Conti, "Rumint – open source network and security visualization tool," http://www.rumint.org.

[10] P. Minarik and T. Dymacek, "Netflow data visualization based on graphs," in *Proceedings of 5th International Workshop on Visualization for Computer Security (VizSec'08)*, Cambridge, MA, September 15 2008, pp. 144–151.

[11] F. Fischer, F. Mansmann, D. Keim, S. Pietzko, and M. Waldvogel, "Large-scale network monitoring for visual analysis of attacks," in *Proceedings of 5th International Workshop on Visualization for Computer Security (VizSec'08)*, Cambridge, MA, September 15 2008, pp. 111–118.

[12] D. Phan, J. Gerth, M. Lee, A. Paepcke, and T. Winograd, "Visual analysis of network flow data with timelines and event plots," in *Proceedings of Workshop on Visualization for Computer Security (VizSEC '07)*, Sacramento, CA, Octoboer 29 2007, pp. 85–99.

[13] D. Cook and L. Holder, "Graph-based data mining," *IEEE Intelligent Systems*, vol. 15, no. 2, pp. 32–41, March 2000.

[14] L. Getoor and C. P. Diehl, "Link mining: a survey," *ACM SIGKDD Explorations Newsletter*, vol. 7, no. 2, pp. 3–12, December 2005.

[15] M. A. Porter, J.-P. Onnela, and P. J. Mucha, "Communities in networks," *Notices of the American Mathematical Society*, vol. 56, no. 9, pp. 1082–1097 & 1164–1166, 2009.

[16] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *Journal of Graph Algorithms and Applications*, vol. 10, no. 2, pp. 191–218, 2006.

[17] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla, "New perspectives and methods in link prediction," in *The 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Washington DC, July 2010, pp. 243–252.

[18] J. O'Madadhain, J. Hutchins, and P. Smyth, "Prediction and ranking algorithms for event-based network data," *ACM SIGKDD Explorations Newsletter*, vol. 7, no. 2, pp. 23–30, 2005.

[19] M. J. Rattigan and D. Jensen, "The case for anomalous link discovery," *ACM SIGKDD Explorations Newsletter*, vol. 7, no. 2, pp. 41–47, 2005.

[20] X. Wan, E. Milios, N. Kalyaniwalla, and J. Janssen, "Link-based anomaly detection in communication networks," in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT '08)*, 2008, pp. 402–405.

[21] Q. Liao, A. Striegel, and N. Chawla, "Visualizing graph dynamics and similarity for enterprise network security and management," in *Proceedings of the 7th International Symposium on Visualization for Cyber Security (VizSec'10)*, Ottawa, Ontario, Canada, September 14 2010, pp. 34–46.

[22] D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: The condor experience," *Concurrency and Computation: Practice and Experience*, vol. 17, no. 2-4, pp. 323–356, February-April 2005.

[23] U. Brandes, M. Eiglsperger, J. Lerner, and C. Pich, *Handbook of Graph Drawing and Visualization (Discrete Mathematics and Its Applications)*. Chapman & Hall/CRC, April 2010, ch. Grraph Markup Language (GrraphML).

[24] H. Bunke, P. Dickinson, M. Kraetzl, and W. Wallis, *A Graph-Theoretic Approach to Enterprise Network Dynamics (Progress in Computer Science and Applied Logic (PCS))*. A Birkhäuser Boston book, 2007.

[25] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846–850, Dec. 1971.

[26] T. Cox and M. Cox, *Multidimensional Scaling, Second Edition*. Chapman & Hall/CRC, September 2000.