Context-Aware Insider Threat Detection

Alex Memory, Henry G. Goldberg, Ted E. Senator
{memoryac, goldberhg, senator} @saic.com
SAIC, Inc.
4001 N. Fairfax Dr., Suite 700
Arlington, VA 22203

Abstract
We are researching ways to detect insider threats in computer usage data crossing multiple modalities – e.g., resources and devices used, network and communication patterns – and where signals of possible threat are highly contextual – e.g., detectable only after inferring user roles, peer groups, collaborators and personal history. The contexts are also dynamic – reflecting a user’s rapid shifts in focus when working on different tasks and longer term changes in interests – and take place in a setting that is identity-aware but privacy-preserving. Although currently focused on the insider threat domain, the architecture, representations and algorithms we are developing are broadly applicable and can lead to interesting future research directions for context-aware computing.

1 Insider Threats and Anomaly Detection
Insiders within an organization, i.e. those with legitimate access to sensitive internal data and systems, can pose a disproportionate threat to the organization and to society if they use this access for malicious purposes (Cappelli, Moore, and Trzeciak 2012). Automatic insider threat detection is challenging because insiders have more knowledge than outsiders of the sensors and defenses meant to prevent malicious use and use it to cover their tracks in novel ways. Also, insiders can hide a few harmful activities in plain sight by interspersing them amongst their normal activities (which they maintain to avoid raising suspicion). Handling this means teasing apart separate activities across time, resource usage, device usage and networks of communication. They can also use multiple user accounts, e.g. administrator accounts, to make identifying them more difficult. The consequence of this, and the fact that cases of insider threat – although costly – are few in number, is that techniques such as supervised machine learning cannot be applied directly.

Anomaly detection, which finds outliers in some context, is a promising alternative for insider threat detection – and also fraud detection and network intrusion detection – where the anomalies are the focus, rather than the overwhelming majority of the data, which represents harmless user activity (Aggarwal 2013; Chandola, Banerjee, and Kumar 2009). However, with this approach it is critical to construct contexts around individuals. By contexts we mean the linking of users’ traits, roles and activities to relevant baseline populations and baseline time periods to compare against. Useful contexts simultaneously reveal threats and minimize false positives. To do this we are developing representations of activities of users and have techniques for designing relevant baselines for specific activity scenarios.

Our existing research has led to the PRODIGAL (Proactive Detection of Insider Threats with Graph Analysis And Learning) architecture, and the results we have achieved so far (Young et al. 2013; Senator et al. 2013), done in a privacy-preserving setting with a real corporate database of computer use activity, suggest promising new directions for research for context-aware computing.

2 An Architecture for Insider Threat Detection
The PRODIGAL architecture finds suspected insider threat instances using models of user activity at multiple levels. Dependencies between observable events and latent activity states are captured on a blackboard model, and components for anomaly detection are specified with a specialized visual language.

Blackboard Model
We are developing our system’s high-level representation based on the blackboard model (Corkill 1991). Reasoning about insider threat activities traverses levels of the blackboard model bottom-up from observed low-level events to infer threats and top-down from hypothesized threats for confirmation in observed events, activities, or behaviors; the levels are shown in Table 1. Atomic transactions and other observations such as file access events and emails reside in the event level and are selected for use based on measures of data quality. Extents are users and groups over time periods and are used to calculate features, e.g., counts of emails between collaborators, to discover activities in the next level,

This work was done by a team of researchers from SAIC, Georgia Tech, Oregon State University, University of Massachusetts and Carnegie Mellon University.
which include both explicit indicators, such as a rise in negative sentiment, and unexplained anomalies.

Next, behaviors, such as copying data or shifting to a new position, are interpretations of activities and are the highest-level elements in the model that are still observable; goals, such as IP theft, and threats are latent, with the former matched to known scenarios from case studies and the latter calculated in the context of the organization’s risk profile.

Components for reasoning within the model are specified using Anomaly Detection Language, which we describe next.

**Anomaly Detection Language**

In anomaly detection, we detect anomalies associated with some entity, which may be an individual insider, i.e. system user, or the anomalies may be associated with a group of entities, so we adopt the more general entity extent. Similarly, the anomaly may be associated with a particular period of time, known as a temporal extent, and the combination of these two is an extent. The inputs to our analysis are records of (trans)actions by entities comprising values of fields known as features, and the outputs are scores on the extents, with higher scores possibly meriting further investigation. Additionally, because the analysis is done in stages, scores themselves are treated as features. We use this vocabulary in a specialized visual language called Anomaly Detection Language; we describe its syntax next.

**Syntax** Components in the language, cf. Fig. 1, are rectangles connected by lines representing sets of records passed between them and have types, such as statistical anomaly detector type (denoted by the symbol $S$), group detector ($G$) which discovers communities of entities which can be used as baseline populations, classifier ($C$) which filters and partitions input records, aggregator ($A$) which summarizes records into features, normalizer ($N$) which rescales records and features with respect to some context, and or ($\lor$), and and ($\land$) used when sets of records are joined and contain different values for the same feature, and union ($\cup$) and intersection ($\cap$) when no combinations are necessary.

The inputs to the component enter on its left, with the input time period placed below the input line, and the optional baseline drawn as a second line on the left with the baseline population and time period placed above and below that line, respectively. The output entity and temporal extents are super- and sub-scripts on the component type, respectively and exit on the component’s right that, when joined, represents a join of the records, in the sense that tables of a database are joined.

If a baseline is provided, a baseline type specifies how the baseline is used by the component. In a cross-sectional baseline ($C$) entity extents are compared to others within the same temporal extent; in a longitudinal baseline ($L$) each entity will be handled individually and different temporal extents for that entity are compared to one another; and a simultaneous baseline ($S$) combines the first two and compares each input extent to all baseline extents. Table 2 lists the expected behavior of an anomaly detector with a variety of inputs and baselines. For example, if the input user data we are analyzing are from the month of November and the baseline population against which we are comparing is all other users from the same month, then the anomaly detector will score each user in November, i.e. each user-month, which makes this a cross-sectional baseline.

Whenever a component may output more than one output class of records, e.g., a binary classifier has (+) and (−) output classes, they should be placed to the right of the component inside circles connected to output lines, unless only one class of output is needed and that class is clear from context, in which case the output class can be omitted. Weights are scalars in the unit interval used to transform features – usually scores – and are drawn as the letter $w$ inside a rectangle. The type of weighting should be put in a description above the rectangle. Finally, the output of the system is drawn as the letter $O$ inside a circle.

We detect insider threats with the language by specifying a relevant context as features and baselines, which we illustrate with examples next.

**Basic Example** Consider a small example in the language. In Fig. 2a, we find anomalous users based on the number of blacklist web sites they visit. We (a) retrieve all the URL access records, (b) keep only accesses to URLs on a blacklist with $C_{URL}$, (c) count the number of such accesses for each user for each month with $C_{user}^{\text{max}}$, and (d) run a statistical anomaly detector over all users in the month using the
Table 1: Levels of the blackboard model, hypotheses inferred in the levels, contexts used for that reasoning, and examples of what resides in the levels.

<table>
<thead>
<tr>
<th>Level</th>
<th>Hypotheses</th>
<th>Contexts</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threat</td>
<td>Belief</td>
<td>Corporate risk analysis</td>
<td>Risk of data theft, risk of sabotage, risk of exposure to carelessness, risk of unauthorized access</td>
</tr>
<tr>
<td>Goal</td>
<td>Plans/Scenarios</td>
<td>Case studies</td>
<td>Theft of IP, hiding level of file access, masquerading, exploring system, circumventing security, carelessness</td>
</tr>
<tr>
<td>Behavior</td>
<td>Tasks</td>
<td>Domain ontology</td>
<td>Copying data, unhappy/worried, reassigned/new project, change of access rights</td>
</tr>
<tr>
<td>Activity</td>
<td>Indicators/Anomalies</td>
<td>Peer groups, user history</td>
<td>Unusual file activity level, unusual email activity, change work schedule/habits, rise in negative sentiment</td>
</tr>
<tr>
<td>Extent</td>
<td>Identities/groups</td>
<td>Historical norms</td>
<td>Access counts, removable drive counts, email counts, groups, sentiment</td>
</tr>
<tr>
<td>Event</td>
<td>Data quality</td>
<td>System health</td>
<td>File events, emails, logon/off, instant messages, process start/stop</td>
</tr>
</tbody>
</table>

Table 2: The expected behavior of an anomaly detector with a variety of inputs and baselines.

<table>
<thead>
<tr>
<th>Input Time Per.</th>
<th>Population</th>
<th>Time Period</th>
<th>Type</th>
<th>Baseline</th>
<th>Anomaly Detector</th>
<th>Extent</th>
<th>Scores Each…</th>
</tr>
</thead>
<tbody>
<tr>
<td>November</td>
<td>All users</td>
<td>November</td>
<td>Cross-sectional</td>
<td>User-Month</td>
<td>User in November</td>
<td>User-Day</td>
<td>User in November</td>
</tr>
<tr>
<td>November</td>
<td>All users</td>
<td>November</td>
<td>Longitudinal</td>
<td>User-Day</td>
<td>Day in November for each user</td>
<td></td>
<td></td>
</tr>
<tr>
<td>November</td>
<td>All users</td>
<td>November</td>
<td>Cross-sectional</td>
<td>User-Day</td>
<td>User for each day of November</td>
<td></td>
<td></td>
</tr>
<tr>
<td>November</td>
<td>All users</td>
<td>November</td>
<td>Simultaneous</td>
<td>User-Day</td>
<td>User-day in November</td>
<td></td>
<td></td>
</tr>
<tr>
<td>November</td>
<td>User roles</td>
<td>November</td>
<td>Cross-sectional</td>
<td>User-Month</td>
<td>User in November for each user role</td>
<td></td>
<td></td>
</tr>
<tr>
<td>December</td>
<td>All users</td>
<td>Jan. - Nov.</td>
<td>Simultaneous</td>
<td>User-Month</td>
<td>User in Dec. vs. previous user-months</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Counts of blacklist URL accesses in \( S^{\text{user}}_{\text{ext}} \). Finally, we (e) return anomaly scores for each user. Next, in Fig. 2b we extend the previous example with baseline populations based on the divisions of the organization (b) from user records, e.g., Lightweight Directory Access Protocol (LDAP) (a) in which users are compared cross-sectionally (c).

Saboteur Example In the Saboteur scenario, cf. (Cappelli, Moore, and Trzeciak 2012), shown in Fig. 3, we look for users with administrator access that could be sabotaging (or preparing to sabotage) systems. We count \( A^{\text{user}}_{\text{day}} \#2 \) times when security-related processes, e.g., antivirus, are stopped \((C_{\text{action}}^{\text{#1}}, C_{\text{proc}}^{\text{#1}})\) and high-risk files are modified at atypical intervals \((C_{\text{file-ext}}^{\text{#1}}, C_{\text{action}}^{\text{#2}}, T_{\text{event}}^{\text{#1}})\), add the length of time worked \( A^{\text{user}}_{\text{day}} \#1 \) and whether it was a non-work day \( C^{\text{user}}_{\text{day}} \#1 \), then normalize w.r.t. the users’ history \( N^{\text{user}}_{\text{day}} \).

To limit the baseline to administrators, we find them in LDAP \((C^{\text{user}}_{\text{#1}})\) and combine \((V^{\text{user}})\) that set with those who act like administrators \((C^{\text{user}}_{\text{#2}})\) according to the file types they access \((C_{\text{file-ext}}^{\text{#2}}, A^{\text{user}})\) compared to the users’ work group in email \((G^{\text{user}})\). Finally, we score the user-days \( S^{\text{user}}_{\text{day}} \) and weight days leading up to a user’s departure from the organization more heavily \((C^{\text{user}}_{\text{day}} \#2, V^{\text{user}}_{\text{day}})\).

Results on a Real Corporate Database

As part of the Defense Advanced Research Projects Agency’s Anomaly Detection at Multiple Scales (ADAMS) program (Defense Advanced Research Projects Agency 2010), a database of monitored computer usage activity in an organization with approximately 5,500 people is collected and made available for research purposes in a closed testbed.\(^3\) Data are collected using a tool called SureView\(^\text{TM}\) (Raytheon Company), which is resident on user workstations and captures user actions such as logins, file access, emails and instant messages, printer usage and browser usage (Raytheon Company 2010). All collected data are treated as legitimate activity, a valid assumption given the rarity of malicious insider activity in the organization. An independent red team develops scenarios of insider threat activity and augments this database with instances of such scenarios superposed on (sets of) users whose normal activity corresponds to the background characteristics of users involved in each scenario. The signal-to-noise ratio is approximately 0.2% of users and 0.08% of user-days. Scenarios are made available to researchers monthly, with answer keys consisting of identifiers of the artificially malicious users and descriptions of the scenario activities provided only after detection results have been generated. Each month’s data consists of approximately 1,000 actions per day per user, or about 5.5 million records per day. All iden-

---

\(^3\)The database is from a large corporation whose identity is not allowed to be disclosed publicly. All data are used with permission in a closed facility subject to all necessary privacy protections.
tifying and personal data are anonymized by hashing to unique identity numbers to preserve privacy.

In Fig. 4 we share preliminary results on the ADAMS data from applying our approach for contextual anomaly detection at the activity level (4a) and at the goal level (4b). The activity-level detector, which uses the repeated impossible discrimination ensemble algorithm, achieves high overall accuracy with an area under the ROC curve of 0.98. In contrast, the goal-level detector run on similar data has the effect of concentrating matching threat instances at the top of the results list while ignoring others. This trade-off between a low-level detector’s broad applicability and a high-level detector’s higher confidence but narrower focus is what we are working to overcome by reasoning at multiple levels with the blackboard model, such that we achieve higher performance and robustness than the detectors individually.

**Conclusion**

We are researching ways to detect insider threats in computer usage data crossing multiple modalities where signals of possible threat are highly contextual and dynamic and in a setting that is identity-aware but privacy-preserving. Although currently focused on the insider threat domain, the architecture, representations and algorithms we are developing are broadly applicable and can lead to interesting future research directions for context-aware computing.

**Acknowledgments**

Funding was provided by the U.S. Army Research Office (ARO) and Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-11-C-0088. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

**References**


Raytheon Company. 2010. SureView™ proactive endpoint information protection.
