Identifying Compromised Users in Shared Computing Infrastructures: a Data-Driven Bayesian Network Approach

Antonio Pecchia[1], Aashish Sharma[2], Zbigniew Kalbarczyk[2], Domenico Cotroneo[1], Ravishankar K. Iyer[2]
[1]Dipartimento di Informatica e Sistemistica, Università degli Studi di Napoli Federico II
via Claudio 21, 80125, Naples, Italy
[2]Center for Reliable and High Performance Computing, University of Illinois at Urbana-Champaign
1308 W. Main St., 61801, Urbana, IL, USA
{antonio.pecchia, cotroneo}@unina.it, {aashish, kalbarcz, rkiyer}@illinois.edu

***Abstract*—The growing demand for processing and storage capabilities has led to the deployment of high-performance computing infrastructures. Users log into the computing infrastructure remotely, by providing their credentials (e.g., username and password), through the public network and using well-established authentication protocols, e.g., SSH. However, user credentials can be stolen and an attacker (using a stolen credential) can masquerade as the legitimate user and penetrate the system as an insider.**

This paper deals with security incidents initiated by using stolen credentials and occurred during the last three years at the National Center for Supercomputing Applications (NCSA) at the University of Illinois. We analyze the key characteristics of the security data produced by the monitoring tools during the incidents and use a Bayesian network approach to correlate (i) data provided by different security tools (e.g., IDS and NetFlows) and (ii) information related to the users' profiles to identify compromised users, i.e., the users whose credentials have been stolen. The technique is validated with the real incident data. The experimental results demonstrate that the proposed approach is effective in detecting compromised users, while allows eliminating around 80% of false positives (i.e., not compromised user being declared compromised).

Keywords - security; credential stealing; intrusion detection; correlation; Bayesian network.

I. INTRODUCTION

The growing demand for processing and storage capabilities posed by large-scale scientific and business applications, has led to the deployment of high-performance computing infrastructures [1]. This trend is going to persist in the near future, as shown by the deployment of new supercomputing facilities, e.g., BlueWaters [7], and the wide adoption of new service paradigms, such as, the *cloud computing* [8]. Protecting the integrity and the confidentiality of data and applications executing on these infrastructures from unauthorized accesses is of paramount importance.

Users log into the computing infrastructure remotely, by providing their credentials (e.g., username and password), through the public network and using well-established authentication protocols, e.g., SSH [2]. However, user credentials can be stolen using phishing or social engineering techniques and made available to the attackers via the cyber-security market [10] [18]. By using stolen credentials, an attacker can masquerade as the legitimate user and penetrate the system essentially as an insider.

An access to the system performed with stolen credentials is hard to detect and it may lead to serious consequences, such as the attacker obtaining root-level privileges on the machines of the system or breach of privacy, e.g., email access. Therefore, the timely detection of ongoing suspicious activities is crucial for secure system operations. For these reasons, computing infrastructures are currently equipped with multiple monitoring tools (e.g., intrusion detection systems (IDS) and file integrity monitors) allowing system administrators to detect suspicious activities. However, the need for ensuring high coverage in detecting attacks, calls for accurate and highly sensitive monitoring, which in turn leads to a large number of false positives. Furthermore, the heterogeneity and the large volume of the collected security data makes it hard for the security team to conduct timely and meaningful forensic analysis.

This paper deals with credential stealing incidents, i.e., incidents initiated by means of stolen credentials, which occurred during the last three years at the National Center for Supercomputing Applications (NCSA)[1]. In our earlier study [4] we showed that credential stealing is the top security violation, involving around 26% of all the incidents occurred at the NCSA. The paper proposes an approach to automate the investigation of the security data in identifying *compromised* users, i.e., the users whose credentials have been stolen and used by an attacker to penetrate and to misuse the system.

We analyze the key characteristics of a subset of security alerts collected during the days the incidents occurred and use a Bayesian network approach to correlate (i) data provided by different security tools (e.g., IDS and NetFlows) and (ii) information related to the users' profiles to identify a compromised user. The key findings of the study are:

- *The effectiveness of the alerts to detect compromised users widely varies.* Some alerts are highly accurate and triggered only when a user is actually compromised. Others generate a significant number of false positives. For instance, the *watchlist* alert is observed twice in the analyzed data set and, in both cases, it points to compromised users. In contrast, *HotClusterConn* (a file downloaded by a node of the infrastructure that is not supposed to do this) alert is

[1] www.ncsa.uiuc.edu - University of Illinois at Urbana-Champaign

potentially raised by about 64 users per day; however, most of these users are not actually compromised.

- *Correlating multiple alerts via statistical techniques, e.g., a Bayesian network approach is an effective way to enable accurate detection of compromised users.* Using the Bayesian network as a *decision mechanism,* we are able to automatically remove from the analysis around 80% of false positives (i.e., not compromised user being declared compromised) without missing any compromised user.
- *The approach is effective in case of new/unknown incidents.* We used the proposed Bayesian network approach to analyze the data collected during a very recent incident (occurred on Oct-29-2010) at the time the study was conducted. Although the incident is included neither in the training nor in the validation data sets, our approach is able to identify the compromised user.

The paper is organized as follows. Section II presents background and related work in the area. Section III provides the description of the data sources and shows how the investigation of the alerts has been automated. Section IV presents available incident data and provides insights on the effectiveness of the security alerts at identifying compromised users. Section V describes the adopted model and the underlying assumptions. The model is validated by means of the real incident data in Sections VI and VII. Section VIII concludes the work.

II. Background and Related Work

The increasing growth and diversification of the Internet services has led to an *underground economy* trading stolen credentials during the last years [10]. Despite the existence of specific and more sophisticated prevention techniques, e.g., [14], security violations conducted by using stolen credentials represent a challenging threat in several application domain, such as, the Internet banking [11], grid-computing infrastructures [2], and operating systems [12]. In this paper, the problem of identifying *compromised users* is considered in the context of an open networked environment at the NCSA. The extensive use of the SSH protocol to authenticate the users of the NCSA machines is a serious security threat: as a matter of fact, it has been shown that around 26% of a representative set of security incidents occurred during the last five years at the NCSA, has been performed with stolen credentials [4].

Several studies deal with the security issues related to the use of the SSH protocol. For example, [18] describes the use of statistical techniques, based on timing data collected from the network, that can then be used to obtain information about what the users type during SSH sessions. Authors in [2] investigate the pervasiveness of the attacks in case of SSH-based credential stealing. Finally, it has been shown that the information stored by SSH, such as, the `known_hosts` file, can be used to expand an attack to further machines once a machine has been initially compromised [13]. These works provide valuable insights to improve the security of SSH. Nevertheless, it has to be noted that the users might easily lose the control of the credentials [18], particularly when considering recent threats, such as phishing and social engineering techniques. As a result, once the credentials have been stolen, even the most sophisticated authentication mechanisms might not be able to protect from an unauthorized access.

Security analysis has been often conducted by using the data collected via *honeypots*, e.g., [15], [16], or simulated intrusions, such as [17]. In this work, the analysis encompasses real security data collected during *spontaneous* incidents occurred at the NCSA; the observed alerts are correlated with a Bayesian network approach to go back to the potential compromised users.

Bayesian networks have been widely adopted in the area of security analysis with rather diverse purposes. For example, [19] measures the network security risk by introducing the notion of *dynamic* Bayesian networks in order to include temporal factors in the analysis. The authors in [20] use a Bayesian network approach to evaluate the effects of different placements of intrusion detectors within a distributed system. The modeling of the behavior of the attacker via Bayesian networks is proposed in [21]: the authors aim to evaluate the risk level of critical resources. More recently, [22] adopts a Bayesian network model to deal with the *uncertainty* of the stages of security attacks.

Bayesian network approaches, applied to the design of more effective intrusion detection systems, are closer to our work. The strategy described in [23] unifies misuse- and anomaly-based detection to identify both well-known and zero-day attacks. The authors in [24] deal with the analysis of the features of the system calls, such as, string length and character distribution, to detect ongoing attacks. Finally, [25] extends the open-source Snort intrusion detection system by including a further processing stage, based on Bayesian networks, to reduce the number of false alarms.

In this paper, we attempt neither to develop nor to improve an existing low-level detection tool. Instead, we propose a Bayesian network approach, which has been designed on the top of the monitoring tools available at the NCSA infrastructure, to support the system administrators in the task of identifying compromised users.

III. Target System and Security Data

The study encompasses the data on credential stealing incidents collected at NCSA at 2008-2010 timeframe. Credentials stealing incidents can occur in virtually any network that is accessible from the Internet (e.g., social networking sites, email systems, corporate networks allowing VPN access) or from an intranet or a business network managed by an IT department within a corporation. For this reason, the key findings of this study can be used to drive the design of better defensive mechanisms in organizations other than the NCSA.

The NCSA computing infrastructure consists of about 5,000 machines (including high-performance clusters, small research clusters and production systems, such as mail and file servers) accessed by worldwide users. Fig. 1 provides a

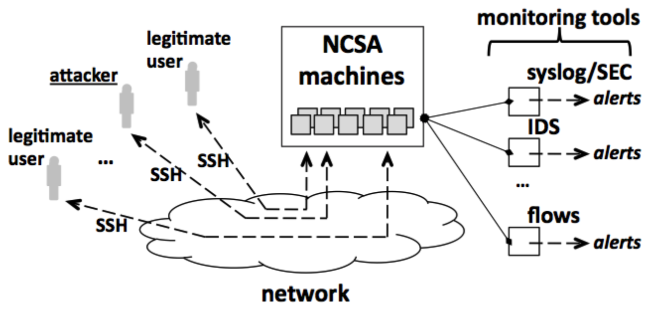

Figure 1. Overview of the target system.

high-level overview of the target system. Users log into the system *remotely*, by forwarding their credentials through the public network via the SSH protocol. Credentials might have been stolen. Thus, an attacker can masquerade as the legitimate user and penetrate the system – *the user is compromised*. The security monitoring tools deployed in the NCSA network infrastructure are responsible for: (i) alerting about malicious activities in the system, e.g., a login from a blacklisted IP address or the occurrence of suspicious file downloads and (ii) collecting (in security logs) data relevant to detected incidents. An in-depth analysis of the security logs can then be used to identify compromised users.

In the following, we describe the alerts produced in case of incidents involving credential stealing (selected according to results in [4]). The process of relating the alerts to the potential compromised users is also discussed.

A. Data and alerts

The log produced by the target machines, collected with the `syslog` protocol, is used to identify the users, which access the infrastructure. To this aim, we extract from the system logs the entries reporting the occurrence of a remote login:

```
month day hh:mm:ss NODE sshd[*]: Accepted AUTH_MEC
for USER from IP_ADDR port PORT ssh2,
```

Note that the login might have been performed with stolen credentials and thus, the entry in log may represent the starting point of a security incident.

The `NODE` field denotes the machine of the infrastructure reporting the message. `AUTH_MEC` is the authentication mechanism used to perform the login, e.g., *password* or *gssapi-with-mic*. `USER` is the username chosen by the legitimate owner of the account and `IP_ADDR` denotes the external address the logging comes from. The analysis of these fields is valuable to deal with incidents conducted with stolen credentials. For example (as it will be detailed in Section III-B), the timestamp and the `NODE` information can be used to relate the alerts raised by the security tools to the users that *potentially* triggered the alerts. Furthermore, each time a specific user logs into the system, `NODE`, `AUTH_MEC`, and `IP_ADDR` are stored in a database, with the aim to create the user profile based on the history of his/her past connections to the system.

In the following, we describe the alerts produced by the monitoring tools. Due to space limitations, the focus is on the alerts triggered in case of incidents conducted with stolen credentials [4]. An *id* has been associated to each alert and used throughout the remaining of this paper. First, we discuss alerts that can be triggered when the login violates the user profile. The detection of potential violations is done by checking login and profile data against rules set in Simple Event Correlator (SEC) [3]:

- *unknown address* (A1): login comes from a previously unknown IP address, i.e., the user never logged from that IP according to his/her profile;
- *multiple login* (A2): the same external IP address is used by multiple users to log into the system;
- *command anomaly* (A3): a suspicious command is executed by the user;
- *unknown authentication* (A10): according to the profile data, the user has never logged into the system by using that authentication mechanism;
- *anomalous host* (A11): the login is reported by a node within the infrastructure that has never been used by the user;
- *last login > 90 days* (A12): the last login performed by the user occurred more than 90 days before the current one.

In most of cases, the occurrence of a profile alert, *alone*, does not provide the definitive proof that the user has been compromised. For example, A1 is raised each time a user (legitimate or not) logs for the first time from a remote site that is not stored in the profile. In order to increase the chance of correctly detecting compromised users, the analysis of profile alerts is combined with the data provided by the security tools, e.g., IDS and NetFlows (Fig. 1), available at the NCSA network infrastructure. In the following, we describe the security alerts used in the detection process:

- *HotClusterConn* (A4): a node of the computing infrastructure performs a download, although it is never expected to execute this action. An additional alert (A13) is introduced to indicate that the downloaded file exhibits a *sensitive* extension, e.g., `.c`, `.sh`, `.bin`, `.tgz`;
- *HTTP* (A5) and *FTP* (A9) *Sensitive URI*: these alerts are triggered upon detection of well-known exploits, rootkits, and malwares;
- *watchlist* (A7)*:* the user logs from a *blacklisted* IP address; the list of suspicious addresses is hold and distributed among security professionals;
- *suspicious download* (A14): a node of the computing infrastructure downloads a file with a *sensitive* extension.

Finally, two further alerts, i.e., (A6) and (A8) have been designed by combining profile and security data. Alert (A6) is generated whenever the remote IP address used to perform a login is involved in subsequent anomalous activities, such as a suspicious file download. Similarly, the alert (A8) is generated if a user responsible for a multiple login is potentially related to other alerts in the security logs.

It should be noted that an event could trigger more than one alert. For example, the download of a file with a sensitive extension, performed by a node that is not supposed to download any file, can trigger the alerts A4, A13, and A14. While the occurrence of a profile alert leads to an initial level of *suspiciousness* about the user, a set of subsequent notifications, such as command anomalies or suspicious downloads, might actually represent the symptoms of an ongoing system misuse. Correlating multiple data sources is valuable to improve the detection capabilities and ruling out potential false alarms.

B. *Automating the analysis of the alerts*

The timely investigation of the alerts is crucial in identifying compromised users and initiating proper recovery actions. However, the analysis can become a time-consuming activity because of the need to correlate alerts coming from multiple sources. In order to automate the alerts analysis we developed a software tool[2] to: (i) parse the content of heterogeneous security logs and (ii) produce a more suitable representation of the security data for facilitating the Bayesian network approach.

Given the data logs (both syslogs and logs produced by the security tools) the tool returns a **user/alerts table,** which provides: (i) the list of users that logged into the system during the time the logs have been collected and (ii) a set of 14-bit vectors (one for each user), with each bit of the vectors representing one of the alerts introduced in Section III-A. Given a user, a bit in the vector assumes value `1` if at least one alert of that type (observed in the security log) has been *potentially* triggered by that user. In order to illustrate the concept, Fig.2 shows a hypothetical user/alerts table. For example, a binary vector of [**1**00**1**000000**1**000] is associated with *user_1,* which indicates that during the observation period, the *user_1* was potentially responsible for triggering three alerts: *unknown address* (A1), *HotClusterConn* (A4), and *anomalous host* (A11).

In the following, we describe the procedure (implemented in the tool) to infer the user that potentially triggered an alert that is observed in the security log. The procedure is relatively simple for the **profile alerts**. For example, the *unknown address* and the *command anomaly* alert have the following formats, respectively.

```
SEC: no IP entry for user USER in the database
```

```
User USER attempting to execute command 'uname -a'
COMMAND on command line.
```

In both cases, `USER` denotes the specific user who triggered the alert. In particular, the command `uname -a` is flagged as *anomalous*, since it is often used by attackers to obtain information about a node, e.g., the kernel version and the hostname. The obtained information can be subsequently used to execute an exploit.

[2] The tool runs under the Linux operating system and consists of a set of `bash` scripts.

alerts / users	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14
user_1	1	0	0	1	0	0	0	0	0	0	1	0	0	0
user_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
…														
user_N	0	0	0	0	0	0	1	0	0	0	0	1	0	0

Figure 2. Example of user/alerts table.

In practice, not all alerts provide enough contextual information to determine the responsible user. Let consider an example,for the *HotClusterConn* alert

```
t=1274736822 HotClusterConn sa=SRC_IP sp=58281/tcp
da=DST_IP dp=80/tcp p=80/tcp method=GET url=[…].
```

In this example, the *timestamp* and the *originating node* are given by the **`t=`** and **sa=** fields, respectively. While we know that the alert is triggered on May-24-2010 at 23:33:42, it is difficult to pinpoint the user responsible for the alert.

In such cases, we adopt a time-based approach to identify a *set of users* that potentially triggered the specific alert. More precisely we: (i) extract the *timestamp* and the *originating node id* from the alert data, (ii) identify (based on the data in the log of the node that raised the alert) all users that logged in to that node in the interval `[t-Δ; t]` (where `t` denotes the timestamp of the alert and Δ is a fixed time window), and (iii) flag as *suspicious* all users identified with this procedure as potentially responsible for this alert. The time window Δ is assumed 3 hours in the context of this work. The analysis of the data shows that an attacker is likely to initiate the misuse in a relatively short time after he/she breaks in to the system. A time window of 3 hours is thus sufficient to relate the alerts to the potential compromised user. When applying this procedure to our example, we flag as *suspicious* all users that logged on the node with id **`SRC_IP`** in the interval [20:33:42; 23:33:42].

IV. OVERVIEW OF THE DATA

The study encompasses the data related to 20 security incidents initiated by using stolen credentials and occurred during the last three years, i.e., 2008-2010, at the NCSA. The NCSA security team comprehensively investigated each incident. The key findings of the investigations are summarized in [4]. The *ground truth*, i.e., the information about the actually compromised users, the reports describing the system misuse, and the proposed countermeasures, is available for each incident considered in this study. This detailed knowledge allows validating the effectiveness of the proposed approach after the design. Out of the 20 incidents, 3 have been detected after a third-party notification, i.e., someone outside the NCSA indicated the occurrence of anomalous activities, and 1 incident was a *false positive* (it was erroneously concluded that the user had been compromised). These four *borderline* cases are discussed in Section VII.B.

In order to characterize the ability of each alert A_i ($1 \leq i \leq 14$) to detect compromised users, we estimate the average

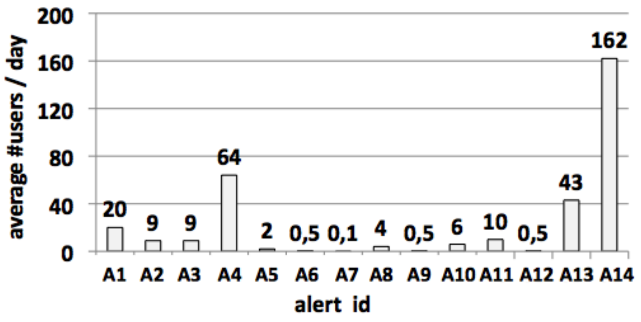

Figure 3. Average number of users (per day) flagged as potentially responsible for each alert A_i

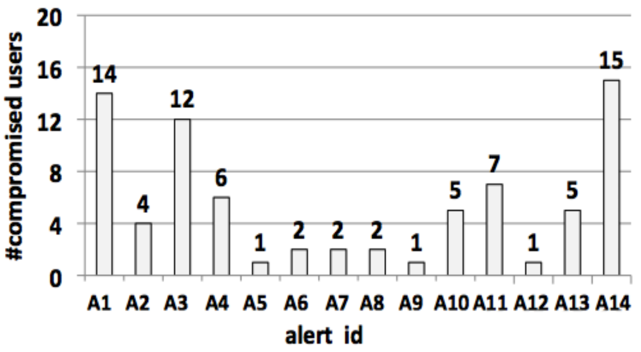

Figure 4. Number of compromised users, which trigger an alert A_i

number of users per day, that are flagged as potentially responsible for A_i. This measure is assessed by analyzing the logs collected during the day the incident occurred. The logs have been processed with the tool presented in Section III-B to obtain the *user/alerts tables*. Fig. 3 reports the average number of users per day flagged as potentially responsible for an alert considered in our analysis. Note that there is a significant variability in the occurrence frequency of different alerts. For example, the *watchlist* (A7) alert is observed for only 2 users during the entire observation period (which spans the 16 incidents analyzed in this study). In both cases, the user was actually compromised. In the context of our analysis, the *watchlist* is considered as a reliable alert, since it has a small number of occurrences (0.125 users/day) and no false positives. On the other end of the spectrum is *HotClusterConn* (A4) alert, which has high occurrence frequency (64 users/day) and relatively high false positive rate (most of the users, flagged as responsible for this alert, are not actually compromised).

The *detection* capability is another important feature of the alerts. The detection capability is assessed by extracting (from the user/alerts tables generated for the 16 analyzed incidents) the 14-bit vectors for each compromised user. Recall that the actually compromised users (i.e., the ground truth) are known for each incident. There are a total of 20 compromised users in the incident data. Fig.4 shows how many compromised users (y-axis) are responsible for the specific alert types (x-axis). Comparison of this data with the one presented in Fig.3 indicates that the alerts with the largest number of potentially responsible users are also likely to be observed when the user is actually compromised. For example, around 20 users per day trigger an *unknown address* (A1) alert (see Fig. 3); however, while most of these alerts turn out to be false positives, in 14 out of 20 cases (as reported in Fig. 4) an actually compromised user triggers this alert. Similarly, the *HotClusterConn* (A4) alert leads to many false positives; nevertheless, it is likely to be triggered by compromised users (6 out 20 cases).

The fact that alerts are often inconclusive could suggest that it might be hard to identify compromised users based on the alert data available in the security log. However, our analysis shows that an actually compromised user is related to more than one alert. In our study, in the average, 3 unique alerts are related to a compromised user for each analyzed incident, such as the joint occurrence of the *unknown address*, *command anomaly*, and *HotClusterConn* alert. Consequently, it is feasible to correlate multiple alerts by means of statistical techniques, in order to discriminate between cases where unreliable alerts can be just discarded, from the ones where an alert provides a stronger evidence for the user to be compromised.

V. PROPOSED APPROACH

Security logs are processed using the tool introduced in Section III-B to produce: (i) the list of users who logged into the system during the entire time interval the logs are collected and (ii) the bit vector reporting alerts potentially related to a given user. Using this information, our objective is *to estimate the probability of a user to be compromised*.

A. The model: Bayesian Network

A data-driven Bayesian network [5], [6] approach is proposed to facilitate identification of compromised users in the target infrastructure. A *Bayesian network* is a direct acyclic graph where each node of the network represents a *variable* of interest in the reference domain. The network allows estimating the probability of one or more hypothesis variable(s) given the evidence provided by a set of information variables. In the context of this work, the *hypothesis variable* is "the user is compromised", while the *information variables* are the alerts related to the user.

We model the problem using a *naïve* Bayesian network, i.e., a single hypothesis node is connected to each information variable. It is assumed that no connections exist among the information variables. The structure of the network is shown in Fig.5. A *naïve network* estimates the probability of the hypothesis variable by assuming the independency of the information variables. In other words, given a user, the presence of an alert in a bit vector does not affect the probability of observing the other alerts. A set of vectors (the ones composing the training set described in Section V-B) is used to validate the assumption on independence of information variables. For each combination of the alerts (A_i, A_j) (for i, j∈ {1, 2, 3, …, 14}), we count how many vectors in the training set contain A_i, A_j, or *both* A_i A_j. Then, the *chi-squared* test [9] is applied to verify the null hypothesis H_0, i.e., *the pair of alerts, A_i and A_j, is independent*. Out of the 91 possible combinations of

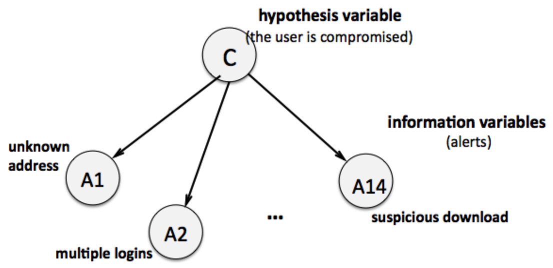

Figure 5. Adopted network

alert pairs, in 28 cases (around 30% of the combinations) the chi-squared test (with 95% significance level) indicates that H_0 has to be rejected. Regardless, for majority of the cases the assumption of alerts independence still holds and hence, it is reasonable to adopt a naïve Bayesian network in the context of our analysis.

It has to be noted that, assuming the independence for a pair of dependent alert, leads to the *overestimation* of the statistical evidence provided by the joint occurrence of the alerts. For this reason, by neglecting the dependencies among the information variables, it might happen that the a-posteriori probability of the hypothesis variable "the user is compromised" is greater than the real value. Consequently, the assumption of alerts independence makes the analysis more conservative; however, more importantly, it does not compromise the detection capability of the adopted Bayesian network.

B. *Training of the Bayesian network*

We use the logs collected in a subset, i.e., 5 out of 16, of the available incidents as training set of the network. Let T denote the adopted training set. T consists of 717 users (and corresponding bit vectors) and encompasses 6 compromised users. T contains the *minimum* number of training incidents, ensuring that almost all the alerts are covered by the actually compromised users, as shown in Fig.6 (*comp_i* denotes a compromised user). The strategy taken in this study has two main advantages (i) it allows analyzing the performance of the Bayesian network with conservative assumptions, i.e., only few training incidents are considered, and (ii) it does not bias the results, because all the adopted alerts are represented by the training set. Although other criteria might have been adopted, this selection of the training set is reasonable to conduct a preliminary analysis of the performance of the proposed approach.

alerts / users	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14
comp_1				1									1	1
comp_2	1		1							1	1			
comp_3		1		1				1					1	1
comp_4	1					1								
comp_5				1			1							1
comp_6					1									1

Figure 6. Alerts related to the compromised users for the incidents in the training set

TABLE I. STRUCTURE OF THE CPT FOR EACH ALERT AI

	compromised (C)	
alert (A_i)	*true*	*false*
true	P(Ai|C)	P(Ai|¬C)
false	P(¬Ai|C)	P(¬Ai|¬C)

The training stage allows tuning the network parameters, i.e., (i) the *a-priori probability* of the hypothesis variable and (ii) the *conditional probability table* (CPT) for each information variable A_i. The a-priori probability of the hypothesis node C (Fig.5) is estimated as P(C) = 6/717 = 0.008. Calculating CPTs needs additional effort. Four parameters must be estimated for each alert as shown in Table I. For example, P(Ai|C) denotes the probability that an alert of type A_i is related to the user, given the user is compromised. Similarly, P(Ai|¬C) represents the probability that an alert of type A_i is related to the user, given the user is *not* compromised.

The probability values of the CPTs are estimated as follows. The overall number of users in the training set T is divided into two disjoint subsets: *good* and *compromised*. Let G and C denote the two subsets, respectively. Note that |T| = |G| + |C|. For each alert Ai, let Li be the set of users (good or compromised) of the training set T, exhibiting that type of alert, e.g., all the users in T responsible for a *command anomaly* alert. P(Ai|C) is the ratio |Li ∩ C| / |C|, i.e., the cardinality of the intersection of Li and C divided by the cardinality of C. Similarly, P(Ai|¬C)=|Li ∩ G|/|G|, i.e., the cardinality of the intersection of Li and G divided by the cardinality of G. P(¬Ai|C) and P(¬Ai|¬C) are the complement to 1 of P(Ai|C) and P(Ai|¬C), respectively.

Table II summarizes obtained results. It can be noted that P(Ai|C) assumes a relatively *high* value, which depends on the number of compromised users included in the training set T. Furthermore, P(Ai|¬C) is a measure of the quality of the alerts, in terms of the number of false positive they are likely to raise. For example, there is a high chance for a not compromised user to be responsible for an *unknown address* (A1) or *HotClusterConn* (A4) alert. Similarly, the chance to observe a *watchlist* (A7) alert if the user is not compromised is extremely low. These probability values confirm the results discussed in Section IV.

TABLE II. VALUES OF P(AI|C) AND P(AI|¬C) COMPUTED FOR EACH ALERT OF THE NETWROK

	A1	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**
P(Ai|C)	0.333	0.166	0.166	0.500	0.166	0.166	0.166
P(Ai|¬C)	0.042	0.022	0.012	0.303	0.021	0.001	0.001
	A8	**A9**	**A10**	**A11**	**A12**	**A13**	**A14**
P(Ai|C)	0.166	0.001	0.166	0.166	0.001	0.500	0.833
P(Ai|¬C)	0.019	0.001	0.011	0.012	0.001	0.240	0.527

By means of the proposed Bayesian network, given a user and the related vector of alerts, it is possible to perform the following query, i.e., "*What is the probability* ***P(C)*** *for the user to be compromised given the user is responsible for 0 or more alerts"*. In the following, we analyze how P(C) varies across the incidents and investigate the possibility of using the network as the *decision tool*.

VI. Analysis of the Incidents

The effectiveness of the proposed approach is assessed using the incident data, which have not been used to train the network. The list of the incidents is given in the first column of Table III. The analysis consists in computing the probability P(C) for each user logged into the NCSA machines during the day the given incident occurred. Table III (the last column) gives the P(C) values computed for the actually compromised users (recall that the ground truth is known from the data). The results of this analysis allow selecting a *classification threshold*, i.e., a value for P(C) to discriminate compromised from not compromised users.

A. Sample incidents

Example 1: **incident #5** (Table III), occurred on Jul-24-2009. During the day the incident occurred, 476 users logged into the system. P(C) is estimated for each bit vector (i.e., user) in the user/alerts table computed for this incident. Fig.7 gives the histogram of the number of users with respect to the observed P(C) values. For the majority of the users (410 out of 476) the computed P(C) is 0.02%. Closer look into data reveals that none of these users is responsible for alerts observed in the security logs and hence, we can conclude those users are *not compromised*.

In all the other cases, i.e., 66 out of 476, at least one alert (in the security log) can be associated with each user. These users are considered as *potentially compromised*. For example, during the day the incident occurred, 33 users triggered a *multiple login* alert. However, 24 of these alerts were false positives caused by a training class using the GPU cluster at the NCSA – all users logged from the same IP address. Consequently, the presence of the *multiple login* alert alone does not provide a strong enough evidence of a user to be compromised and the Bayesian network returns a small value of P(C) (i.e., 0.21%). The compromised user in the incident #5 exhibits a value of P(C) around 28.54% (dotted arrow in Fig.7). In this case, an *unknown address*, a *multiple logins*, and a *command anomaly* alerts are associated with the user.

We also observed cases where a large value of P(C) is computed for not compromised user. For example, a user who jointly triggered an *unknown address*, *multiple login*, *unknown authentication*, and *anomalous host* alert, results in a probability value of 87.73%. Nevertheless, it has to be noted that the number of false indications produced by the Bayesian network is small. In the incident #5, only for three users the P(C) value is greater than the one computed for the actually compromised user (to the right of the dotted arrow in Fig.7). In this incident, the proposed approach brings the number of manual investigation of potentially compromised users down from 66 to 4.

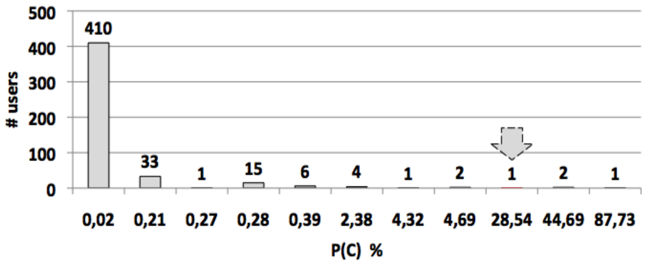

Figure 7. Example 1: histogram of the number of users for the observed P(C) values (incident #5, occurred on Jul-24-2009)

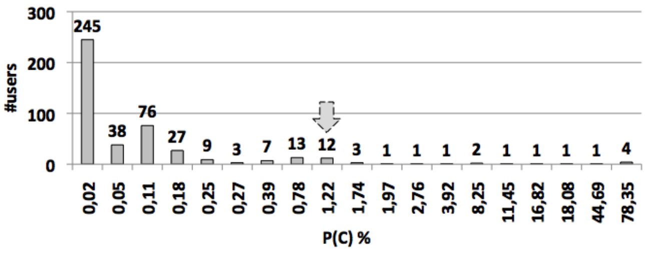

Figure 8. Example 2: histogram of the number of users for the observed P(C) values (incident #7, occurred on Apr-22-2009)

Example 2: **incident #7** (Table III), occurred on Apr-22-2009. During the day the incident occurred, 446 users logged into the system. Fig. 8 gives the histogram of the number of users for the observed P(C) values. No alert is associated with 245 users (P(C) = 0.02%). In this incident, the security logs contain 22 notifications of *suspicious download*, i.e., 22 files with a *sensitive* extension are downloaded by some of the machines in the NCSA infrastructure. These alerts result in significant number of potentially responsible users (76 as shown in Fig. 8). Again, according to the training data, this alert *alone* does not provide strong enough evidence for a user to be compromised (P(C) = 0.11%). The P(C) for the user compromised in the incident #7 is 1.22% (dotted arrow in Fig. 8), which is a much smaller value as compared with the Example 1 (P(C) = 28.54%). Only two alerts – *unknown address* and *suspicious download* – are observed for the compromised user. These alerts are likely to generate many false positives as discussed in Section IV (Fig.3). Furthermore, closer look into data reveals that the same two alerts (*unknown address* and *suspicious download)* are also raised for 11 out of 12 *not compromised* users. As shown in Fig. 8, if P(C) = 1.22% is used as a threshold to identify compromised users, we end up with 16 false detections i.e., not compromised users would be flagged as compromised (all cases to the right of the dotted arrow in Fig. 8). As in the previous example, large P(C) (e.g., 78.35%) is observed for some of users even if they are not actually compromised.

B. Discussion

The described analysis is conducted for each incident in the validation set. The analysis results reported in Table III reveal that P(C), for different incidents, varies within a large

TABLE III. List of the incidents and P(C) values observed for the compromised users

incident ID	incident date	#compromi-sed users	P(C)[a]
1	May-03-2010	1	16.8%
2	Sep-08-2009	1	99.3%
3	Aug-19-2009	2	13.5%; 13.5%
4	Aug-13-2009	1	4.3%
5	Jul-24-2009	1	28.5%
6	May-16-2009	1	0.7%
7	Apr-22-2009	1	1.2%
8	Nov-03-2008	1	28.5%
9	Sep-07-2008	3	99.7%;99.9%;76.8%
10	Jul-12-2008	1	44.6%
11	Jun-19-2008	1	18.1%

a. Assumed by the compromised users

TABLE IV. Analysis of the incidents with the Bayesian network approach

incident		#susp	TH. [≥0.7]		TH. [≥1.2]		TH [≥4.3]	
ID	#users		#actn.	*ratio*	#actn.	*ratio*	#actn.	*ratio*
1	305	252	41	0.16	32	0.12	19	0.07
2	477	122	29	0.23	27	0.22	14	0.11
3	353	312	134	0.42	39	0.12	33	0.10
4	309	203	34	0.16	28	0.13	13	0.06
5	476	66	11	0.16	11	0.16	7	0.10
6	491	7	1	0.14	*0*	*0*	*0*	*0*
7	446	201	41	0.20	37	0.18	*10*	*0.05*
8	447	251	62	0.24	50	0.20	43	0.17
9	497	422	137	0.32	85	0.20	58	0.13
10	193	118	9	0.07	3	0.02	3	0.02
11	380	280	3	0.01	3	0.01	2	0.01
avg.	***398***	***221***	***50***	***0.20***	***29***	***0.12***	***19***	***0.07***

range of values. One can see that in all, but two cases (incidents #6 and #7), P(C) for the compromised users is relatively large. For the incident #6, a single alert, *HotClusterConn,* (with a sensitive extension) is associated with the compromised user. However, this alert, if observed alone, is quite unreliable. As shown in Fig.3 (A13), around 43 users per day are potentially responsible for this type of alert. Our Bayesian network returns very small value for P(C) = 0.7%.

As discussed, most of the *suspicious* users, i.e., the ones related to at least one alert, are not actually compromised: P(C) is generally small. This finding suggests that the Bayesian network can be used to remove the *noise (false positives)* induced by the alerts. In other words, *it is feasible to define a classification threshold, i.e., P(C) that will allow suppressing a significant fraction of false positives while still identifying all compromised users*.

VII. Supporting Decision with the Bayesian Network Approach

In this section the proposed Bayesian network is used to discriminate compromised from not compromised users by means of a *classification threshold*: *if the alerts related to a user result in a value of P(C) greater than the classification threshold, we assume that the user is compromised.*

A. Analysis of the incidents

According to the results provided in Table III, the *minimum* classification threshold that allows detecting all the compromised users is 0.7% (this is the value of P(C) observed for the compromised user in the incident #6). This value is used to quantify the effectiveness of the Bayesian network approach.

Table IV summarizes obtained results. *Column 1* reports an incident in the validation set (the ID of the incident is the same as Table III). For each incident, *column 2* reports the number of users that logged into the system during the day the incident occurred and *column 3* provides the number of *suspicious* user, again, the ones related to at least one alert. *Column 4* gives the number of *actionable* users, i.e., the subset of the suspicious users whose probability of having been compromised is ≥ 0.7%. For each incident, the ratio between the number of actionable and suspicious users quantifies the effectiveness of the proposed approach at reducing the number of false indications/positives. The ratio is reported in *column 5*. For example, in the *worst* case (represented by the incident #3, where 134 out of 312 suspicious users are actionable) the tool removes around 58%, ((1 - 134/312)*100%) of false indications. As shown in the last row of Table IV, the average ratio (estimated across all the incidents) is around 0.20 for the classification threshold set to 0.7%. In other words, for the analyzed data set *the Bayesian network approach automatically removes around 80% of false positives (indicating an uncompromised user to be compromised).*

The effectiveness of the proposed strategy is bounded by the need for selecting a relatively *low* threshold. Actually, 0.7% is a conservative value, which allows avoiding false negatives, i.e., missing actually compromised users. We analyze how the number of actionable users varies when the value of the classification threshold increases. Results are reported in *columns 6 and 7* and *columns 8 and 9* of Table IV, for the threshold values 1.2% and 4.3%, respectively. According to Table III, 1.2% and 4.3% are the next two smallest P(C) values observed in the analysis. The compromised user of the incident #6 is undetected when the threshold is 1.2%. However, the network removes around 88% of false positives. Similarly, when the classification threshold is set to 4.3%, the compromised users of the incident #6 and #7 are undetected. In this case, the network removes around 93% false positives.

Analysis results suggest also that the Bayesian network approach can help the system administrators by directing toward the users that are likely to be compromised. After the security logs are collected, the tool can be used to obtain the list of the users exhibiting a particularly large value of P(C), e.g., all the users whose probability of being compromised is ≥4.3%. This procedure eases the task of the administrators. As indicated in the last row of Table IV, in average, around 19 users out 398, i.e., only 4% of all the users, which log into the NCSA during a normal day of operation, surpass the 4.3% classification threshold. In the case when an actually compromised user is not detected (i.e., the classification threshold is set to large value), the analyst can decrease the threshold in order to augment the set of possible suspicious users to be investigated.

B. *Analysis of the borderline cases*

We also assess the effectiveness of the network for *borderline* cases, such as, incidents notified by third-party, or *normal* days, i.e., days where no incidents occurred. The analysis is conducted by using 0.7% as the classification threshold. Obtained results are summarized in Table V. The meaning of columns 2,3,4,5 is the same as in Table IV. Furthermore, for each case (if applicable) *columns 6 and 7* report the maximum observed P(C) and P(C) for the compromised user, respectively. The main findings of the analysis are discussed in the following.

External notifications. Three of the analyzed incidents are missed/undetected by the NCSA monitoring tools and are discovered because of notifications by external sources e.g., third party. Our tool is used to analyze the logs collected during the days the undetected incidents occurred, and, for each incident, the bit vectors in the user/alerts table are queried against the network. All compromised users are undetected with the proposed approach. The values of P(C) for the compromised users are 0.02% (external notif. #1 and #2) and 0.4% (external notif. #3), which are below the assumed classification threshold of 0.7%. In the two former cases, no alerts in the log seem to be generated by the compromised users. In the latter case, the only *command anomaly* alert is observed. This alert, if observed alone, does not provide strong evidence that the user is really compromised. Since the Bayesian network approach relies on the low-level monitoring infrastructure, when *no alert is triggered*, it is not feasible to identify a compromised user.

False positive. On Nov-03-2008 the NCSA security team is alerted for a login performed by a user that triggers an *unknown address*, a *command anomaly*, an *unknown authentication*, and an *anomalous host* alerts. The user/alerts table obtained from the logs collected during that day confirms the joint occurrence of these alerts. The computed value of P(C) is 92.9% and hence, it is reasonable to assume that the user is compromised. System administrators contacted the owner of the account, which confirmed his/her activity and that the login was legitimate.

These cases enforce our earlier finding that alert correlation improves the ability of identifying compromised

TABLE V. ANALYSIS OF THE BORDERLINE CASES WITH THE BAYESIAN NETWORK APPROACH

event			**TH. [≥0.7]**		**P(C) max**	**P(C)** [a]
type (date)	**#users**	**#susp**	**#actn.**	*ratio*		
ex. notif. #1 Apr-21-2009	386	176	26	0.15	4.6%	0.02%
ex. notif. #2 Mar-18-2009	269	179	63	0.35	96.3%	0.02%
ex. notif. #3 Feb-09-2009	289	28	3	0.11	1.7%	0.4%
false positive Nov-03-2008	447	251	62	0.24	92.9%	92.9%
norm. day #1 Jun-30-2010	323	227	25	0.11	87.7%	-
norm. day #2 Jul-25-2010	154	88	28	0.32	41.7%	-
new incident Oct-29-2010	358	159	32	0.20	65%	9.4%

a. Assumed by the compromised users (if applicable)

users; however, the deficiencies of the low-level monitoring infrastructure (missing events or *false* notifications) can mislead analysis results.

Normal days. We query the network with the user/alerts tables obtained with the logs collected during 2 normal days of operation, Jun-30-2010 and Jul-25-2010 (see Table V). The number of the *actionable* users, i.e., the users whose probability of being compromised is ≥0.7%, is small (25 and 28, respectively). It can be noted that some users exhibit a high P(C). For example, during the normal day #1, P(C) is 87.7% for a user, which jointly exhibits an *unknown address*, a *multiple login*, an *unknown authentication*, and an *anomalous host* alert. Again, the proposed strategy reduces the number of false indications due to not trusted alerts, however, if the user is potentially responsible for multiple alerts, P(C) is high even if the user is not actually compromised.

New incident. We analyze the data collected during a recent incident occurred on Oct-29-2010 at the time the study was conducted. The incident is included neither in the training nor in the validation data set of the network. During the day the incident occurred, 358 users logged into the NCSA machines. Among them, 159 users raised alerts. The Bayesian network approach allows reducing the initial set of 159 suspicious users to 32 actionable users. The compromised user is correctly included in the actionable set and detected (three alerts, i.e., *unknown address*, *command anomaly*, and *HotClusterConn* are raised by the compromised user: P(C) = is 9.4%).

VIII. CONCLUSION

The paper proposes a Bayesian network approach to support the detection of compromised users in shared computing infrastructures. The approach has been validated by means of real incident data collected during the last three years at the NCSA. The results demonstrate that the Bayesian network approach is a valuable strategy to drive the investigations efforts of the security team. Furthermore, it is able to significantly reduce the number of false positives (around 80% with respect to the analyzed data). We also observed that the deficiencies of the underlying monitoring tools could affect the effectiveness of the proposed network.

Future work will encompass the analysis of other categories of security incidents not only the credential stealing. To this aim, the network will be enhanced by introducing additional alerts. Furthermore, we will use this experience to design an *on-line* tool implementing the proposed strategy: streams of security alerts will be processed *on-the-fly* and enable timely recovery actions as soon as a reasonable evidence of an ongoing misuse is observed.

ACKNOWLEDGMENT

This work has been partially supported by the project “Overcoming Security Market Obstacles for SME's Involvement in the Technological Supply chain" (OSMOSIS, http://www.osmosisecurity.eu, GA-242416) in the context of the EU Seventh Framework Programme (FP7), by the Italian Ministry for Education, University, and Research (MIUR) within the Project of National Research Interest (PRIN) “DOTS-LCCI: Dependable Off-The-Shelf based middleware systems for Large-scale Complex Critical Infrastructures”, by NSF grant CNS 10-18503 CISE, the Department of Energy under Award Number DE-OE0000097, the Air Force Research Laboratory and the Air Force Office of Scientific Research, under agreement number FA8750-11-2-0084 (The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon), IBM Corporation as part of OCR (Open Collaboration Research), and Boeing Corporation as part of ITI Boeing Trusted Software Center.

REFERENCES

[1] http://www.top500.org - TOP 500 List.

[2] J. Zhou, M. Heckman, B. Reynolds, A. Carlson, M. Bishop, “Modeling network intrusion detection alerts for correlation”, ACM Transactions on Information and System Security, 10(1), 2007.

[3] R. Vaarandi “SEC – A Lightweight event correlation tool”, in Proc. of IEEE IPOM’02, 2002.

[4] A. Sharma, Z. Kalbarczyk, J. Barlow, R. Iyer, “Analysis of Security Data from a Large Computing Organization”, in Proc. of the International Conference on Dependable Systems and Networks, 2011, 506-517.

[5] F. Jensen, “Bayesian Networks and Decision Graphs”, Springer, New York, USA, 2001.

[6] J. Pearl, “Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference”, Morgan Kaufmann, 1997.

[7] http://www.ncsa.illinois.edu/BlueWaters - National Center for Supercomputing Applications, Blue Waters project.

[8] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, “A view of cloud computing”, Commun. ACM 53, 4, April 2010, 50-58.

[9] W. Mendenhall, T. Sincich, “Statistic for the Engineering and Computer Sciences”, 2nd ed., San Francisco, CA, Dellen, 1988.

[10] T. Holz, M. Engelberth, F. Freiling, “Learning More About the Underground Economy: a Case-Study of Keyloggers and Dropzones”, Reihe Informatik TR-2008-006, University of Mannheim, Germany.

[11] R. Oppliger, R. Rytz, T. Holderegger, “Internet Banking: Client-Side Attacks and Protection Mechanisms”, IEEE Computer Security, 42(6), June 2009.

[12] F. Weimer, “New openssl packages fix predictable random number generator” (http://lists.debian.org/debian-security-announce).

[13] S.E. Schechter, J. Jung, W. Stockwell, C. Mclain, “Inoculating SSH Against Address-Harvesting Worms”.

[14] N. Haller, “The S/KEY One-Time Password System”, in Proc. of the Internet Society Symposium on Network and Distributed Systems, 1994..

[15] N. Provos, “A Virtual Honeypot Framework”, USENIX Security Symposium, USENIX, 2004.

[16] M. Cukier, R. Berthier, S. Panjwani, S. Tan, “A Statistical Analysis of Attack Data to Separate Attacks”, in Proc. of the International Conference on Dependable Systems and Networks, 2006, 383-392.

[17] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, S. Stolfo, “A generic framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data”, in Data Mining for Security Applications, Kluwer, 2002.

[18] B. Schneier, “Two-factor authentication: too little, too late”, Communications of the ACM, 48(4), 2005.

[19] M. Frigault, L. Wang, “Measuring Network Security Using Bayesian-Network-based Attack Graphs”, STPSA 2008.

[20] G. Modelo-Howard, S. Bagchi, G. Lebanon, “Determining Placement of Intrusion Detectors for a Distributed Application through Bayesian Network modeling”. RAID, September 2008.

[21] R. Dantu, P. Kolan. “Risk management using behavior based bayesian netowrks”. IEEE Int.l Conf. on Intelligence and Security Informatics, May 2005.

[22] P. Xie, J.H. Li, X. Ou, P. Liu, R. Levy, “Using Bayesian Networks for Cyber Security Analysis”, in Proc. of the International Conference on Dependable Systems and Networks, 2010, 211-220.

[23] P.G. Bringas, “Intensive use of Bayesian belief networks for the unified, flexible, and adaptable analysis of misuses and anomalies in network intrusion detection and prevention systems”, in Proc. of the 18th Int.l Conf- on Database and Expert Systems Applications, 2002.

[24] C. Kruegel, D. Mutz, W. Robertson, F. Valuer, “Bayesian Event Classification for Intrusion Detection”, Proc. ACSAC 03, Dec. 2003.

[25] W. Tylman, “Misuse-based intrusion detection using Bayesian Networks”, in Proc. of the 3rd Int.l Conference on Dependability of Computer Systems DepCoS-Relcomex, 2008.