Collaborating as normal: detecting systemic anomalies in your partner

Olgierd Pieczul[1,2] and Simon N. Foley[2]

[1] Ireland Lab, IBM Software Group, Dublin, Ireland.
olgierd@pieczul.net
[2] Department of Computer Science, University College Cork, Ireland.
s.foley@cs.ucc.ie

Abstract. It is considered whether anomaly detection techniques might be used to determine potentially malicious behavior by service providers. Data mining techniques can be used to derive patterns of repeating behavior from logs of past interactions between service consumers and providers. Consumers may use these patterns to detect anomalous provider behavior, while providers may seek to adapt their behavior in ways that cannot be detected by the consumer. A challenge is deriving a behavioral model that is a sufficiently precise representation of the consumer-provider interactions. *Behavioral norms*, which model these patterns of behavior, are used to explore these issues in a on-line photograph sharing style service.

1 Introduction

In today's digital world, individuals and organizations perform much of their computing and communications using third party services. These service *consumers* and *providers* interoperate according to their own, possibly conflicting, requirements. For example, an individual consumer uses a social media service provider to communicate with friends: the service is free, however, the consumer may wish to minimize advertisments/loss of privacy, while the provider may wish to maximize advertising revenue by weakening consumer privacy. Similarly, the provider of a public cloud infrastructure may be willing to risk degraded consumer service for the sake of additional consumer revenue, while consumers seek certain service agreements. Consumers and producers rely on each other to behave accordingly, however each have to recognize that it may be in the interest of the other to cut across their requirements.

In this paper we explore how consumers might detect malicious provider behavior that is at variance with consumer requirements, and how malicious providers, might in turn, adapt their behavior in ways that cannot be detected by the consumer. This malicious provider behavior is not a conventional Dolev-Yao style external attacker [5, 13] since the consumer relies on the provider's 'normal' behavior. Nor is the behavior that of a insider-attacker [4,6] that is to be mitigated by security controls within the provider. We characterize this behavior as that of a *systemic* attacker: it is the provider itself that is the attacker. A

systemic attack may result from the deliberate intentions of a provider or arise from an incompetent provider that itself has been compromised in some way.

In principle, a consumer could use a reference monitor to check provider interaction against policies of acceptable behaviors. In practice, however, the scale and complexity of the systems involved mean that it is not reasonable to expect a complete and coherent specification of such behaviors, regardless of the consumer's understanding of the requirements. A proactive consumer might use browser-based security controls [11] in an attempt to prevent Cross Site Scripting attacks coming via an incompetent provider, write some network-packet controls in effort to block unwanted content, rely on protocols such as OAuth [9] to control access, or even use task-based polices [15] to control provider interaction sequences. Such consumer-side controls on provider interaction will likely be ad-hoc and incomplete, focusing on behavior perceived to be critical, with an assumption that other activities, known or unknown, are not significant. However, it is often the side-activities that can lead to security concerns.

We argue that log data of past interactions between consumer and provider(s) can be used to derive policies of acceptable behavior and that the consumer can use anomaly detection techniques to monitor provider compliance. When a consumer is unable to (fully) articulate their expectation of a provider, then the consumer should be interested in knowing when provider behavior deviates from what are considered 'normal' interactions of the past. This deviation may be an indication of a security concern. Such system log mining techniques have been used elsewhere to infer acceptable behavior/policies for anomaly detection [7,12] process mining [1–3] and security policy mining [8,10]. In this paper we consider how the system log mining techniques described in [12] might be used by a service consumer to discover models for these 'normal' interactions and which can be used to monitor for potential systemic attacks by service providers.

The paper is organized as follows. Section 2 sketches the operation of a simple online photograph sharing service. Section 3 outlines how a behavioral norm model might be generated from a consumer's log of their interaction with this service. Sections 4 and 5 explore how this behavioral norm model might be used to detect anomalies in single and collaborating provider services. While this paper is exploratory, Section 6 outlines how behavioral norms have been evaluated in practice. Section 7 concludes the paper.

2 An online photograph sharing service

Consider an on-line photograph hosting and sharing service. The service allows users to upload and store their photographs, establish a network of friends with whom to share photographs, comment on photographs, and so forth. The service also provides activity tracking of the users and their friends. Users can view the actions they have performed (for example, the photographs they uploaded and when), and limited tracking of the actions of other users (for example, accesses and comments on the photographs they share). For example, Figure 1 provides a fragment of a log of such actions that are visible to the user `Frank`.

time	user	context	action	id	extra
2013-11-04 16:53:05	Frank	self	login	-	-
2013-11-04 16:55:21	Frank	self	upload_photo	img23	Holidays 2013
2013-11-04 16:57:55	Frank	self	upload_photo	img24	New bike
2013-11-04 17:01:03	Frank	self	share	img23	Lucy
2013-11-04 17:04:29	Lucy	friend	view_photo	img23	-
2013-11-04 17:05:18	Frank	self	share	img24	Bob
2013-11-04 17:05:19	Lucy	friend	comment	img23	I wish, I was there
2013-11-04 17:21:34	Bob	friend	view_photo	img24	-
2013-11-04 17:22:01	Bob	friend	comment	img24	Nice!
...					

Fig. 1. Partial log from the photo hosting service

This activity data need not necessarily come from a conventional text log. Actions/events may be presented to the consumer by the provider using a web interface or as a feed in some common format such as RSS or ATOM and we assume that a consumer is be able to view the events relevant to its interaction with the provider. Events are comprised of attributes; the events in Figure 1 have attributes that provide `time` of event, `user` name, `action` carried out, and whether the action is carried out by the user viewing the log (the `context` value `self`) a `friend` or `other` user, the image `id`, and any `extra` data.

Studying Figure 1, we see that `Frank` logs-in, uploads two photographs, shares photographs with users `Lucy` and `Bob` who in turn view and comment.

Our goal is to discover a model that represents (provider) behavior from the event log that includes the fragment in Figure 1. Analyzing the log events contiguously/in the order in which they appear in Figure 1 does not provide much insight into the behavioral patterns of the provider. For example, representing the behavior in terms of short-range correlations between events, such as n-grams [7], does not reveal any interesting patterns of behavior.

However, a closer inspection of the log in Figure 1 reveals what appears to be two, interleaving, transaction-like patterns of behavior. In the first, `Frank` uploads a photo img23, shares it with `Lucy` who then views and comments. In the second, the same sequence of actions occur in relation to `Frank` sharing img24 with user `Bob`. This analysis identifies a simple transaction-style behavior in the log fragment:

<upload_photo, share_photo, view_photo, comment_photo>

In identifying these transaction style patterns it is important to distinguish the roles that are played by the different event attributes. Intuitively, the attribute value `action` represents the operation being carried out by the event and this operation is effectively parameterized by the image identifier (target attribute `id`). For the purposes of this paper we choose to ignore the `time` attribute as not playing a role in the behavior of the provider (other than providing event temporal ordering). Further study of the log is required to decide whether the `user`, `context` and `extra` attribute values should play a role in this transaction.

3 Behavioral norms

Behavorial norms [12] represent repeating patterns of behavior at different levels of abstraction that can be discovered from event traces/logs. A search process has been developed [12] can be used to determine the event attributes that represent the operations and parameters for potential norms discovered in the event log. These norms may be represented in various forms, such as a database of n-grams.

Considering the log fragment in Figure 1, the search process discovers a behavioral norm depicted as:

```
<self.upload_photo, self.share_photo, friend.view_photo, friend.comment_photo>
```

This is a transaction-style sequence of actions. The search identifies attributes `context` and `action` values as representing the event operation on common target attribute `id` values while attributes `time` and `extra` are considered to have no discernible effect on behavior. Thus, the log sub-sequence

```
2013-11-04 16:55:21   Frank  self    upload_photo  img23   Holidays 2013
2013-11-04 17:01:03   Frank  self    share         img23   Lucy
2013-11-04 17:04:29   Lucy   friend  view_photo    img23   -
2013-11-04 17:05:19   Lucy   friend  comment       img23   I wish, I was there
```

is a valid instantiation of the above norm, while the sub-sequence

```
...
2013-11-04 16:55:21   Frank  self    upload_photo  img23   Holidays 2013
2013-11-04 17:01:03   Frank  self    share         img23   Lucy
2013-11-04 17:04:29   Lucy   friend  view_photo    img23   -
2013-11-04 17:05:19   Lucy   friend  comment       img24   I wish, I was there
...
```

is not a valid instantiation of the norm as it does not involve a common photograph `id`.

Figure 2 depicts likely behavioral norms that might be discovered if given a complete provider log for Frank. The first norm describes the behavior that

```
1  <self.upload_photo, self.share_photo, friend.view_photo, friend.comment_photo>
2  <friend.upload_photo, friend.share_photo, self.view_photo, self.comment_photo>
3  <friend.upload_photo, self.view_photo, self.comment_photo>
4  <other.connect_request, self.accept_connect_request>
5  <self.connect_request, other.accept_connect_request>
```

Fig. 2. Norms for user's collaboration with photo hosting service provider

can be observed from Figure 1. The other norms represent additional kinds of typical 'normal' behavior, such as Frank viewing photos shared by other users, or connecting with friends.

The norms in Figure 2 represent provider (online service) behavior that could be discovered by a consumer (`Frank`) analyzing his event logs. These 'discovered' norms provide insight into the behavior of the provider. `Frank` and his community usage patterns and configuration, such as privacy settings, are reflected in

these norms. For example, `Frank` uses the service's default privacy policy that considers newly upload photos as private. This requires him to explicitly share every photo before it is viewed by other users. Some of `Frank`'s friends have a similar configuration, and this is reflected in the second norm. Other friends configured their account differently to make all of their uploaded photos visible to their friends or public, by default. This behavior is captured in the third norm, which lacks an explicit sharing operation.

4 Provider Anomalies

Assume that `Frank`'s photo hosting service wishes to attract additional traffic and increase the amount of content that is available to their users. To do this, they decide to change their default application behavior. The change is to make all new content visible to the user's friends by default. Users can still configure the policy explicitly in order to override default behavior. Unaware of the new default setting, Frank continues to use the service and uploads new images. Frank's friends may now see the image instantly, without Frank's explicit action to share. This change is made to only the default behavior of the application. It does not modify application's terms of use nor the privacy policy. Frank still has the right to restrict his content, configure his policy differently, or remove any of his content. While this provider change may be done entirely legally it has a negative effect on Frank's use of the application.

Frank's set of norms may be used to detect this application change. His service provider, after the change, will start generating the logs that cannot be matched to the norms in Figure 2. This unrecognized activity may be considered an anomaly and alert Frank to investigate the change. Performing norm discovery on the new log can reveal that a new norm has emerged:

<self.upload_photo, friend.view_photo, friend.comment_photo>

This anomaly is specific to Frank's interaction with the service. For other users, such as those whose photos are shared with others by default, the change has no impact. For such users, the above norm would already be considered an acceptable norm (based on the analysis of their logs).

5 Anomalies across multiple collaborating providers

Continuing the example, Frank uses an additional service provider: an on-line photograph printing service. Using this service he can order prints for his photographs on-line and have them delivered to the friends and family. The service is integrated with Frank's photograph hosting provider. This is convenient for Frank as he can give the printing site permission to access his photographs and order prints without the need to re-upload. The access delegation can be done using a standard protocol such as OAuth [9]. In a typical scenario, Frank accesses the printing service, and selects his hosting service as the location of images.

The printing service accesses Frank's account and downloads photograph miniatures. Frank selects the photographs that he wants printed and for each of them the printing service, with its delegated authority from the photograph sharing service, downloads the full size image files.

The logs (visible to Frank) from both providers for such a scenario are presented at Listing 3. Log events now originate from two different service providers and this is distinguished by a new event attribute `provider` in the logs. In addition, events for actions performed on behalf of Frank by the printing service provider have a `context` attribute value `prtsvc` in the hosting provider log.

```
PRINT SERVICE (provider=print)                HOSTING SERVICE (provider=host)
time     user  context action      id         time     user  context action        id
-----------------------------------------     -----------------------------------------
19:31:05 Frank self    new_order
                                              19:31:19 Frank prtsvc  list_photos
                                              19:31:20 Frank prtsvc  get_thumbnail img01
                                              19:31:20 Frank prtsvc  get_thumbnail img02
                                              ...
                                              19:31:21 Frank prtsvc  get_thumbnail img08
19:33:41 Frank self    select      img03
19:33:52 Frank self    select      img07
                                              19:34:06 Frank prtsvc  get_fullsize  img03
                                              19:34:08 Frank prtsvc  get_fullsize  img07
19:36:02 Frank self    submit_order
```

Fig. 3. Two producers collaboration

Frank has given the printing service a permission to access his photos. While short-lived permission delegations are possible in schemes such as OAuth, many providers offer long-lived *offline* permissions, which are often requested by the third-party providers [14], irrespective of the dangers. The expected behavior is that the service will only access the photos when Frank places a print order. Technically however, there is no such restriction and the print service may access the photos at any time. Frank can only trust that this service provider will behave properly.

Analyzing the hosting service log in isolation the following norm may be discovered:

```
<prtsvc.list_photos, prtsvc.get_thumbnail, prtsvc.get_fullsize>
```

This norm represents the typical way in which a print service accesses user photographs when interacting with the hosting service. With its delegated permission from Frank, the printing service could decide to download all of Frank's photos in the background without interaction with Frank. This activity will generate a log in the hosting service. Based on the behavioral norm above, however, this activity can be regarded as 'normal'.

Building the behavioral norms from the individual printer service log is insufficient to fully capture the interaction between consumer and the two providers. The norms should be discovered from a single log that aggregates the events from

both service providers. In this case, log operations are characterized in terms of three attributes: `provider.context.action` with a sample norm

<print.self.new_order, host.prtsvc.list_photos, host.prtsvc.get_thumbnail, print.self.select, host.prtsvc.get_fullsize, print.self.complete_order>

This norm captures aggregated behavior of all of the parties collaborating together. Any activity of printing service unrelated to Frank's print ordering will be considered abnormal, as it will not match the norm.

6 Norms in action

This paper explores the use of behavioral norms to help interpret anomalies in the interactions between a consumer with its providers. Previous research [12] evaluated the effectiveness of using behavioral norms to represent emergent behavior from system logs. The evaluation demonstrated that behavioral norms can be discovered in logs from a simulated system and in logs from a real-world enterprise on-line collaboration application. The logs contained relatively low-level system attributes and the norm discovery process identified attributes for event actions and targets for the norm transactions.

In practice, the sequence of operations may not be identical even if two parts of log represent the same behavior. For that reason, norms are represented as patterns that match sequences to certain degree of similarity. This similarity level, if set high, it produces large number of very precise norms. If it is low, model contains fewer, more general norms. During the norm search the suitable similarity level is identified.

In a further experiment, we considered how adverse changes in a system configuration might be detected in terms of changes in norms. The simulated application system in [12] was augmented to include a simple access-control mechanism that governed the operations carried out by users. The resulting behavioral norms for the application system reflected the constraints by the underlying access control system. The simulation was modified to reflect a security flaw whereby the access control policy was disabled and this resulted, as anticipated, in the identification of new application behavioral norms. These norms described new behaviors corresponding to system activity with different then previously recorded access rights. Figure 4 depicts comparison between number of norms (for different levels of similarity) for system before and after the change. These experiments confirmed that behavioral model can be successfully built, and reasoned about, from arbitrary system logs with unknown structure of events. In this paper we used the behavior norms model to help interpret anomalies in service consumer-provider scenario. We are currently exploring how this might be evaluated in practice.

7 Discussion

Consumer security is impacted by the provider services with which it directly or indirectly interacts. Individually, providers may have different motivations in

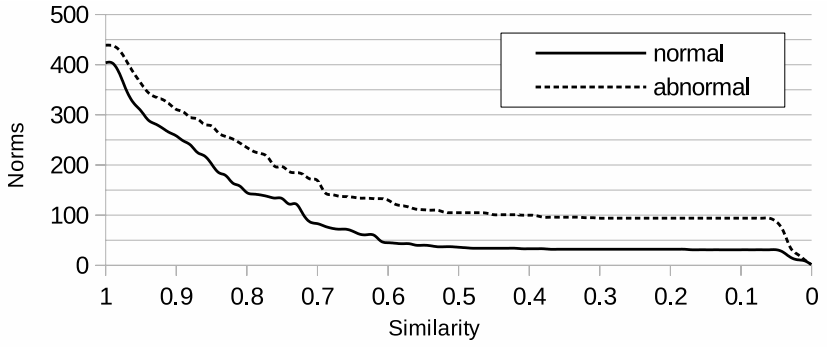

Fig. 4. Number of norms before and after configuration change [12]

providing service and the security mechanisms available to the consumer to control interaction tend to be weak. For example, service providers often provide only coarse grained access controls to their consumers. When multiple applications need to collaborate, they may be given more access than is actually required.

We argue that anomaly detection style techniques can be used by a consumer to monitor interactions with providers. The challenge is to formulate a sufficiently precise model of 'normal' interaction and we propose that consumers mine their provider logs to build models of past, presumably acceptable, behavior. We propose using behavioral norms [12] to model multiple patterns of behavior in a system log.

Conventional anomaly detection is routinely used to help protect a provider from malicious consumers; we have considered using anomaly detection to protect a consumer from multiple, possibly collaborating, providers. A single consumer transaction may span multiple providers interacting with each other and the consumer. Prescribing rules for each of the providers separately is not sufficient. As seen in Section 5, an anomaly may not manifest itself when only single provider-centric rules are considered. The anomaly may be an acceptable activity from the individual provider, but be unacceptable when considered part of a value chain.

Another difficulty in determining normal interaction is distinguishing acceptable and unacceptable provider intertaction. Simply comparing provider behavior against known and precise access control rules is not sufficient. Section 4 illustrated how provider misbehavior can be subtle and within the boundaries of the contract, but is a deviation from normal/past interactions.

If consumers can use behavioral norms to detect malicious provider behavior then a malicious provider might attempt to use the same norms to guide behavior adaptation in ways that cannot be detected by the consumer. This corresponds to a *mimicry* style attack [16], used to bypass anomaly detection systems. For example, in an n-gram based model [7], the attacker crafts an at-

tack sequence that contains malicious code but is built entirely of acceptable (n-gram) sequences.

Investigating whether a malicious provider constrained by behavioral norms would find it difficult to mount a successful mimicry attack is a topic for future research. Behavioral norms provide a model of discovered behavior that is considerably more precise than n-grams. To mimic a behavior, one must consider not only the operation itself, but the other attributes (user and provider in our example) that together represent the actions engaged for a common target attribute value (image `id` in our example) for a given behavioral norm. We conjecture that this provides less flexibility in designing malicious sequences that fit a behavioral norm. Furthermore, as shown in section 5, the model may include aggregated behavior of multiple providers with the consumer. In this case, the malicious provider not only must adjust its own sequences, but must also be able to influence the sequences of the other providers.

Acknowledgments This research has been partly supported by Science Foundation Ireland grant 08/SRC/11403.

References

1. van der Aalst, W.M., Weijters, T., Maruster, L.: Workflow mining: Discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering 16(9), 1128–1142 (2004)
2. Accorsi, R., Stocker, T.: Automated privacy audits based on pruning of log data. In: EDOCW 12th Enterprise Distributed Object Computing Conference Workshops pp. 175–182 (2008).
3. Agrawal, R., Gunopulos, D., Leymann, F.: Mining Process Models from Workflow Logs. Proceedings of the 6th International Conference on Extending Database Technology: Advances in Database Technology, Springer-Verlag (1998)
4. Bellovin, S.: The insider attack problem nature and scope. In: Insider Attack and Cyber Security, Advances in Information Security, vol. 39. Springer (2008)
5. Dolev, D., Yao, A.: On the security of public key protocols. IEEE Transactions on Information Theory 29(2), 198–208 (1983)
6. Foley, S.: A non-functional approach to system integrity. IEEE Journal on Selected Areas in Communications 21(1) (Jan 2003)
7. Forrest, S., Hofmeyr, S.A., Somayaji, A., Longstaff, T.A.: A sense of self for Unix processes. In: IEEE Symposium on Security and Privacy. pp. 120–128 (1996)
8. Frank, M., Buhmann, J., Basin, D.: On the definition of role mining. In: Joshi, J.B.D., Carminati, B. (eds.) ACM Symposium on Access Control Models and Technologies (SACMAT), pp. 35–44. ACM (2010)
9. Hardt, D.: The OAuth 2.0 Authorization Framework. RFC 6749 (Proposed Standard) (Oct 2012), `http://www.ietf.org/rfc/rfc6749.txt`
10. Kuhlmann, M., Shohat, D., Schimpf, G.: Role mining - revealing business roles for security administration using data mining technology. In: Proceedings of the Eighth ACM Symposium on Access Control Models and Technologies. pp. 179–186. SACMAT '03, ACM, New York, NY, USA (2003)
11. Louw, M.T., Venkatakrishnan, V.N.: Blueprint: Robust prevention of cross-site scripting attacks for existing browsers. In: Proceedings of the 2009 30th IEEE Symposium on Security and Privacy. pp. 331–346. IEEE Computer Society (2009)

12. Pieczul, O., Foley, S.: Discovering emergent norms in security logs. In: Communications and Network Security (CNS - SafeConfig), 2013 IEEE Conference on. pp. 438–445 (2013)
13. Ryan, P.: Mathematical models of computer security. In: Focardi, R., Gorrieri, R. (eds.) Foundations of Security Analysis and Design, Lecture Notes in Computer Science, vol. 2171, pp. 1–62. Springer Berlin / Heidelberg (2001)
14. Sun, S.T., Beznosov, K.: The devil is in the (implementation) details: An empirical analysis of OAuth SSO systems. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security. pp. 378–390. CCS '12, ACM, New York, NY, USA (2012)
15. Thomas, R., Sandhu, R.: Task-based authorization controls (TBAC): A family of models for active and enterprise-oriented autorization management. In: Proceedings of the IFIP TC11 WG11.3 Eleventh International Conference on Database Securty XI: Status and Prospects (1998)
16. Wagner, D., Soto, P.: Mimicry attacks on host-based intrusion detection systems. In: Proceedings of the 9th ACM Conference on Computer and Communications Security. pp. 255–264. CCS '02, ACM, New York, NY, USA (2002)