Beat the DIVa - Decentralized Identity Validation for Online Social Networks

Leila Bahri*, Amira Soliman†, Jacopo Squillaci*, Barbara Carminati*, Elena Ferrari*, Sarunas Girdzijauskas†
*University of Insubria, Varese - Italy
*Royal Institute of Technology, Stockholm - Sweden
{leila.bahri, jacopo.squillaci, barbara.carminati, elena.ferrari}@uninsubria.it
{aaeh, sarunasg}@kth.se

***Abstract*—Fake accounts in online social networks (OSNs) have known considerable sophistication and are now attempting to gain network trust by infiltrating within honest communities. Honest users have limited perspective on the truthfulness of new online identities requesting their friendship. This facilitates the task of fake accounts in deceiving honest users to befriend them. To address this, we have proposed a model that learns hidden correlations between profile attributes within OSN communities, and exploits them to assist users in estimating the trustworthiness of new profiles. To demonstrate our method, we suggest, in this demo, a game application through which players try to cheat the system and convince nodes in a simulated OSN to befriend them. The game deploys different strategies to challenge the players and to reach the objectives of the demo. These objectives are to make participants aware of how fake accounts can infiltrate within their OSN communities, to demonstrate how our suggested method could aid in mitigating this threat, and to eventually strengthen our model based on the data collected from the moves of the players.**

I. Introduction

According to measurement studies on social media, there is a substantial existence and influence of fake accounts and bots in the realms of social media platforms. For example, it has been found that at least 7% of accounts on Twitter are created and operated by automated algorithms (i.e., bots), about 20% of social media users are susceptible to accept online friendship requests from unknowns, and approximately one in three users might be deceived by fake accounts thinking they are actually dealing with genuine people [1]. Fake accounts are created for different nefarious goals, such as detouring public opinion, running dishonest advertisement or support campaigns, deceiving target individuals to spy on their activity or steal their personal information, or polluting an environment with biased information with astroturfing[1] intentions [2].

OSN providers have been deploying efforts to fight against fake accounts in their realms. For example, Facebook has recently announced that they work on improving their mechanisms for detecting fake accounts, especially as related to impersonations, such as men who create female profiles to deceive other females, or sex predators who hide behind kind online identities to abuse younger users [3]. Besides, the research community has paid greater effort to the study and the suggestion of techniques to aid the detection of such attacks (as an example see the works in [4] and [5]). Most of these works agree on the fact that fake accounts tend to connect with each other, to form separate communities, and to exhibit behavior that differs them from that of real accounts in terms of usage patterns such as times, amounts, and frequency of interactions. However, adversaries have being sophisticating their techniques and are now, as formulated in [1], trying to get legitimacy from socializing with real users by gaining their trust and becoming their friends.

Typically, fake accounts can be successfully detected as long as they have not infiltrated within communities of real and trusted users. Unfortunately, statistics tell that considerable numbers of OSN users are willing to accept friendships from unknowns [1]. Moreover, OSN users often do not have enough perspective to make informed decisions on the trustworthiness of the new profiles they would be interested in connecting with. This is mostly because the only information they have consists of the profiles public values (as private profile values are not visible before a friendship on the OSN is established) and some general common sense on how a real profile should look like, such as having a real name, a profile picture, and pieces of information that exhibit general cohesiveness.

Motivated from this, we have suggested in [6] a model that exploits community feedback to estimate the trustworthiness of OSN profiles based on their values only. The key idea is to exploit community feedback to detect profiles identity patterns. These are defined as hidden correlations between some profile attribute elements that are representative of the identities of the profiles connected within a given community. Those correlations are then exploited to estimate the trustworthiness of new profiles to a community. To improve this model, we have suggested, in [7], the exploitation of association mining learning techniques, in a decentralized and privacy preserving manner, to unveil these identity patterns within OSN communities. Our Decentralized Identity Validation (DIVa) system has proven reliability in unveiling community identity patterns that can be used to validate profile information with more than 50% increased accuracy compared to evaluating the profile as a whole coherent entity [7]. DIVa operates in three main phases. First, it detects communities in an input OSN graph based on topological structure. Second, it runs its learning algorithm on each community to reveal the correlated attribute sets (CAS) that represent its identity patterns. Finally, it informs all the nodes in every community about these detected CASes. Each node can use these CASes as guidelines to evaluate the truthfulness of new profiles it desires to connect with.

[1]Astroturf refers to fake grass. The term astroturfing has been used to describe the practice of running fake campaigns to create fake grassroots supporters or sponsors for an idea, a product, an organization, etc.

In order to demonstrate the utility of the DIVa system, we propose in this demo a game application, called *beatTheDiva*,[2] that exploits the DIVa strategies against a player who needs to succeed in creating fake profiles that beat the system and establish connections with other real profiles. Starting from a real OSN graph of connected profiles,[3] the DIVa system is first executed to detect topological communities and to learn their CASes. *beatTheDiva* game exploits this input graph to challenge players who are allowed to create their profiles and to try to align them with the identities of their target communities to convince target nodes to accept their friendship requests. To emulate real life scenarios, at the beginning of a game round, players cannot see communities structure and can only see limited information about the profiles in the graph. Players score points with every node they succeed at connecting with against the game's deployed strategies. These earned points can be used to reveal more information about the network. *beatTheDiva* exhibits different game strategies and scenarios to challenge the players. Players who succeed at convincing the majority of nodes in a community to accept their friendship all while maintaining their score balance above a certain threshold win the game. Participants in the demo session will be invited to play the game and try to beat the DIVa. While doing so, players can learn more about DIVa strategies and our approach to validating identities in OSNs. The demo will also help us to validate and improve our DIVa system by analyzing the moves made by players and possible gaps that they could identity and exploit during the game.

The rest of the paper is organized as follows: in Section II we provide an overview of the DIVa system. In Section III we describe the architecture of the suggested game, and in Section IV we illustrate examples of actions and playing scenarios that participants can demonstrate with. Finally Section V concludes the paper and summarizes the main objectives of the demo.

II. DIVA OVERVIEW

In DIVa, we represent an OSN as an undirected graph $G = (V, E)$, where V is the set of nodes (or users) and E is the set of edges (or friendships), where $e_{ij} \in E$ denotes a relationship between nodes v_i and $v_j \in V$. We denote the profile schema adopted in the OSN by $S = \{A_1, A_2, .., A_m\}$. For every node $v_i \in V$, p_i denotes the set of its profile values: $p_i = \{p_i.a_1, p_i.a_2, ..p_i.a_m\}$, where $p_i.a_k$ is the value provided by v_i for $A_k \in S$.

As depicted on Figure 1, DIVa operates in three steps. First, it runs a decentralized community detection algorithm for which the output is modeled as a set of communities $C = \{C_h = (V_h, E_h) | V_h \subset V \text{ and } E_h \subset E\}$. In DIVa, communities might be overlapping with some nodes belonging to more than one community; thus, the boundaries between communities are not deterministic as shown on the example in Figure 1. In the second step, every node learns the local correlated attribute sets it can observe from its direct friends, and in the third step all nodes in a community exchange their local knowledge to agree on the community's level CASes. By the convergence of the third step, all the nodes in a community

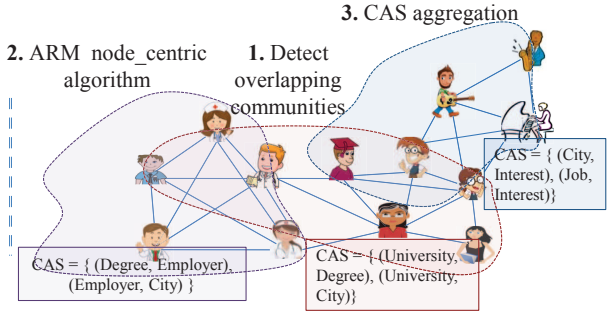

Figure 1. The three phases of the DIVa model [7].

C_h are aware of the community's CASes referred to as the collection $CAS_h = \{cas | cas \subset S\}$, where cas is a CAS in C_h. Nodes are supposed to use this information as guidelines to assess the trustworthiness of new profiles it wants to connect with.

Example 1. Suppose Bob belongs to two overlapping communities C_b and C_p where the CASes are $CAS_b = \{\{city, interest\}, \{job, interest\}\}$ and $CAS_p = \{\{employer, interest\}, \{education, employer\}\}$, for C_b and C_p respectively. Suppose a profile with the name Amy wants to befriend Bob. It is highly probable that Bob's interest in connecting with Amy's profile is shaped by its validation against the identities of the communities Bob belongs to. Therefore, Bob would need to pay special attention to the values Amy's profile exhibits for the corresponding CASes of his two communities C_b and C_p.

III. *beatTheDiva* STRUCTURE

beatTheDiva is a game application that challenges a player to sophisticate her profile against the strategies suggested by DIVa for the validation of OSN identities in a community. Players of the game need to create a profile and try to convince other nodes in the system to befriend them. The nodes in the system deploy DIVa strategies to evaluate the validity of the player profile requesting their friendship.

beatTheDiva emulates real case scenarios where new profiles in an OSN attempt to establish new friendships and to build their trust in the network. In the scenario of a fake profile, there are two possible situations. First, the fake profile is targeting specific individuals by impersonating one of their friends (e.g., a fake account impersonates Alice who is a known friend to a target group of friends). In this case, the fake profile targets specific profiles and hopes to deceive them into accepting her friendship requests. In the second situation, the fake profile is trying to infiltrate within some community without targeting specific individuals. *beatTheDiva* can be applicable to impersonation attacks; however, the focus is more on the second type that deploy tailoring strategies to become members of some target community, hence gaining trust in the network. As such *beatTheDiva* allows players to create a profile and to gradually tailor it, against spending some score points, to infiltrate within some community. Players gain points for every node they succeed at befriending and are able to view her profile values. The game is won when the player establishes enough links to become part of a community.[4]

[2] *beatTheDiva* game app is accessible at: http://strict.dista.uninsubria.it/?page_id=816

[3] We use, as example, the same Facebook datatset as exploited in [7] for experiments.

[4] Community membership is determined based on the community detection algorithm as used in [7].

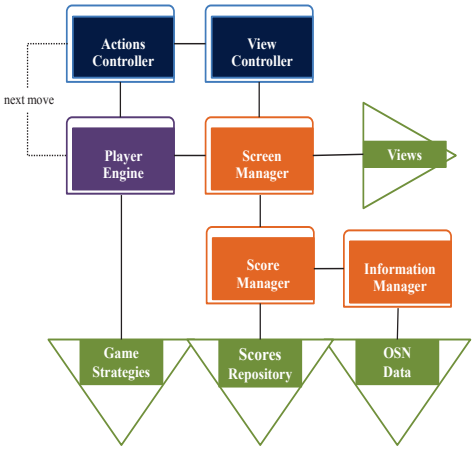

Figure 2. *beatTheDiva* game engine architecture.

Before describing the game scenarios (see Section IV), we first present its architecture and deployed strategies to challenge the players.

A. System architecture and game engine design

Figure 2 shows the general architecture of *beatTheDiva*. The view controller manages the graphical changes on the interface in accordance with the screen manager that prepares the snapshot screens to be displayed in real time. The actions controller listens to the player's actions and communicates them to the player engine that returns the resulting game strategy to the screen manager. This latter collaborates with the score manager to update and display the score of the player. The information manager takes care of revealing more information about the OSN to the player depending on the earned score. In what follows, we detail the strategies deployed by the game engine.

Players create a profile, p_p, according to the schema S adopted in the system, and send friendship requests to chosen nodes in the network.[5] These requests are evaluated based on a simple strategy that accounts for three main factors: 1) *the CAS factor* that is the evaluation of the player against the DIVa CASes in the community of the target node; 2) *the player infiltration factor* that reflects the positioning of the player in the target community; 3) *the behavioral prior factor* that models the behavioral uncertainty that a user in real life would accept or deny a friendship request regardless of its rationality. We define these factors as follows:

Definition 1. ***The CAS factor***. *Let $S = \{A_1, A_2, \cdots, A_k, \cdots, A_m\}$ be the profile schema in the system, where A_k is an attribute name. Let p_p be the profile of the player and p_t, member of community C_t,[6] be the target node by an issued friendship request from p_p. Let F be the set of profiles of the friends of p_t. Let CAS_t be the collection of CASes defined by DIVa in community C_t. Let M_i be the set of profiles from F having the same profile values for $cas_i \in CAS_t$ as p_p. The CAS factor assigned to p_p w.r.t p_t is defined as: $CASF_p(t) = \frac{\sum_{\forall cas_i \in CAS_t} M_i}{|CAS_t|}$.*

To define the *infiltration factor*, we assume that every node v_i in the OSN has a known *influence factor* NI_i that is defined as its clustering coefficient. This latter is a graph based measure that gives an indication on how the region surrounding a node is densely or sparsely connected [8].

Definition 2. ***The infiltration factor***. *Let p_p be the profile of the player and p_t, member of community C_t, be the target node by an issued friendship request from p_p. Let $LF \subset C_t$ be the set of nodes belonging to C_t that the player has successfully befriended. The infiltration factor of the player in community C_t is defined as: $I_p(C_t) = \sum_{\forall v_i \in LF} NI_i$.*

The behavioral prior factor reflects the probabilistic prevision that a node will accept a friendship request. We consider that the more friends a node has, the more prone to accept new friendships it is, especially if its friends are diverse. We base this assumption on a model in graph theory known as preferential attachment [8]. This one suggests that an old node creates an edge with a new one with a probability proportional to its degree. We assume that the average node degree in the input graph and the average community size are known, and we define the following:

Definition 3. ***The behavioral prior factor***. *Let ad be the average node degree and acs be the average community size in the network $G = (V, E)$. The behavioral prior factor of node $v_i \in V$ with degree d_i is defined as $bpf_i = \frac{d_i}{ad * acs}$.*

The game engine evaluates a friendship request based on Procedure 1. A *strategy* is computed as the value of the CAS factor if the infiltration factor is zero (lines 1-2), or as the fraction of the CAS to the infiltration factors of the player w.r.t the target node (lines 3-4). The fraction is considered because the higher the infiltration factor, the more information the player is supposed to have on the community structure; thus, the higher the CAS factor should be. A CAS factor that is lower than the infiltration factor reflects bad choices of the player. The ratio of the CAS factor to the infiltration factor should outweigh the behavioral prior factor of the target node for the request to be accepted (line 5). If it is not, the request is denied (line 6)

Procedure 1 Evaluation of a friendship request by the game engine

Require: $CASF_p(t)$, $I_p(C_t)$, and bpf_t
Ensure: *accepted* or *denied* message
1: **if** $I_p(C_t) = 0$ **then**
2: $\quad strategy \leftarrow CASF_p(t)$
3: **else**
4: $\quad strategy \leftarrow \frac{CASF_p(t)}{I_p(C_t)}$
5: **if** $strategy \geq \frac{1}{bpf_t}$ **then return** *accepted*
6: **elsereturn** *Denied*

IV. Playing *beatTheDiva*

The goal of a player in *beatTheDiva* is to befriend enough friends from a community to become member of it. To achieve this, a player creates a profile, sends friendship requests, scores points for each accepted friendship request (evaluated based on Algorithm 1), accesses profile information of befriended nodes,

[5] We use the terms *player* and *player's profile* interchangeably.

[6] If p_t belongs to more than one community, C_t is the one the player p_p befriended more friends from, if available; otherwise, C_t is randomly selected from all community membership of p_t.

and uses score points to improve her profile presentation based on the network information she gains throughout the game, and/or to send more friendship requests.

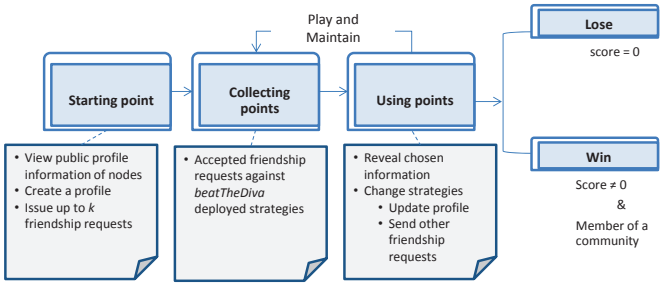

Figure 3. *beatTheDiva* main playing process.

Figure 3 represents the general process of *beatTheDiva*. When starting the game, players can view nodes in the adopted OSN graph[7] and some general profile information from some nodes. This corresponds to the profile information that the nodes set as public in their privacy settings. Players create a profile in accordance with the profile schema adopted by the adopted OSN graph and can send up to k friendship requests, where k is a variable set within the application to limit the initial moves a player can perform. With every friendship request sent by the player and accepted by the game engine, the player earns score points that vary depending on the influence of the newly befriended node in the community and on the infiltration factor of the player's profile in that same community. The player can use these points to make updates to her profile, or to send new friendship requests. The player has access to all the profile information of the node(s) that have accepted its friendship requests. The player loses if the score falls down to a value zero and wins if successfully getting enough friends from a community to become member of it.

A summary of moves that a player can perform in the game is given in what follows:

- **Create a profile:** the player starts the game by creating a profile then views the game's main monitor as on Figure 4.a. The nodes in the graph are displayed without links.
- **Send friendship requests:** the player can view public information of nodes and send up to k friendship requests (k=3 in the example in Figure 4).
- **View response notifications:** the game notifies the player about the refused requests and draws edges between the player's node and the ones that accepted the friendship (see Figure 4.b). The player's score is updated accordingly.
- **View befriended nodes information:** the player can access complete information about the befriended nodes.
- **Spend earned points:** the player can spend score points to send more friendship requests, to update her profile, or to view the influence factor of some chosen nodes.
- **View achieved scores and factors:** the game score is permanently displayed on the monitor. The player can click on their node to view information on achieved factors such as their infiltration factor within communities.

a. John starts the game b. John befriended 3. He can now see her friends

Figure 4. Game monitors of a starting round and after one node is befriended.

V. Conclusion

We suggest in this demo a game that challenges players to create fake profiles that could cheat our DIVa system and infiltrate within some honest community in an adopted real OSN graph. The game uses DIVa strategies that we have proposed to help OSN users reliably estimate the trustworthiness of new profiles desiring to befriend them. From this demo, we aim to raise participants awareness regrading the threat of how fake accounts can gain network trust from deceiving real users, and to demonstrate and analyze our DIVa system in the game's scenarios.

References

[1] J. Taylor. (2015) 7http://oursocialtimes.com/7-of-twitter-users-are-not-human/

[2] T. Simonite. (2015) Fake persuaders. [Online]. Available: http://www.technologyreview.com/news/535901/fake-persuaders/

[3] Z. Miners. (2015) Facebook wants to get better at detecting fake profiles. [Online]. Available: http://www.pcworld.com/article/2893192/facebook-wants-to-get-better-at-detecting-fake-profiles.html

[4] C. Xiao, D. M. Freeman, and T. Hwa, "Detecting clusters of fake accounts in online social networks," in *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*. ACM, 2015, pp. 91–101.

[5] Y. Boshmaf, D. Logothetis, G. Siganos, J. Lería, J. Lorenzo, M. Ripeanu, and K. Beznosov, "Íntegro: Leveraging victim prediction for robust fake account detection in osns," in *Proc. of NDSS*, 2015.

[6] L. Bahri, B. Carminati, and E. Ferrari, "Community-based identity validation in online social networks," in *Proceedings of the 34th international conference on Distributed Computing Systems*. IEEE, 2014.

[7] A. Soliman, L. Bahri, B. Carminati, E. Ferrari, and S. Girdzijauskas, "Diva: Decentralized identity validation for social networks," in *Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on*. IEEE, 2015.

[8] M. E. Newman, "Clustering and preferential attachment in growing networks," *Physical Review E*, vol. 64, no. 2, p. 025102, 2001.

[7]Recall that we adopt in the demo the same Facebook dataset exploited in [7].