Journal of Internet Services and Applications
a SpringerOpen Journal

RESEARCH Open Access

Catching modern botnets using active integrated evidential reasoning

Yongning Tang[1*], Guang Cheng[2,3], James T Yu[4] and Bin Zhang[4]

Abstract

Botnets are now recognized as one of the major security threats to start various security attacks (e.g., spamming, DDoS). Although substantial research has been done towards botnet detection, it is becoming much more difficult today, especially for highly polymorphic, intelligent and stealthy modern botnets. Traditional botnet detection (e.g., signature, anomaly or flow based) approaches cannot effectively detect modern botnets. In this paper, we propose a novel active integrated evidential reasoning approach called SeeBot to detect modern botnets. SeeBot can seamlessly and incrementally combine host and network level evidences and incorporate active actions into passive evidential reasoning process to improve the efficiency and accuracy of botnet detection. Our experiments show that both performance and accuracy of botnet detection can be greatly improved by the active evidential reasoning, especially when the evidence is weak, hidden or lost.

Keywords: Botnet detection; Evidential reasoning

1 Introduction

The total number of computers belonging to botnets increased from 3 millions in April-June 2009 to 6.5 millions during April-June 2010 [1]. Apparently, traditional botnet detection (e.g., signature, anomaly or flow based) approaches [2-6] cannot effectively detect and stop modern botnets. Based on a 2010 poll with chief information security officers and senior IT security directors at Fortune 500 corporations, all respondents stated that they considered malware and botnet to be a serious threat to their enterprise IT security.

Botnets are collections of infected computers that are controlled remotely by cyber-criminals. Originally botnets were created for a specific purpose such as sending spam, identity theft or DDoS (distributed denial-of-service) attacks. However, in 2010 bots that were designed to provide the cyber-criminal with the ability to build designer botnets (e.g., Zeus-based botnets) were rented out to other cyber-criminals for specific purposes (spam, identity theft, DDoS, etc). These criminal organizations invest significant building logical groupings of compromised systems that are organized around a sophisticated, resilient Command-and-Control (C&C) infrastructure or even through social networks [7]. Such criminal networks are exceptionally stealthy and easily evade signature or behavior-based defenses. They can mimic normal application and traffic patterns, and can change their core software far faster than traditional security solutions can update their signature-based systems. We refer to such professionally designed and cyber-criminal oriented botnets as modern botnets, which have the following features:

- Highly polymorphic: The characteristics of botnets are varying even faster (e.g., via polymorphism or code obfuscation) than the signature update from security vendors. The prevalence of improved do-it-yourself (DIY) botnet construction kits and associated exploit packs make this feature much more evident in 2010 [8].
- Highly intelligent: The bots, used to be called zombies, are powered with much more intelligence now, such as Honeypot-aware botnets [9]. Modern botnets can run multiple simultaneous infection mechanisms, update the malware installed on their victims systems regularly, and optimize their serial variant malware production systems to release "personalized" and one-of-a-kind malware with each new victim infection.

*Correspondence: ytang@ilstu.edu
[1] School of Information Technology, Illinois State University, Normal, IL 61790, USA
Full list of author information is available at the end of the article

- Highly stealthy: Because of commerical motivation, modern botnets are designed to be more stealthy via many different mechanisms, such as randomly selected ports, traffic encryption and peer-to-peer based C&C.

Substantial research work has been done towards botnet detection. However, most botnet detection approaches attempted to follow and catch the trend of new botnet design, and then develop certain understanding or assumptions about the corresponding botnets. For example, (1) assuming certain botnet has a trackable payload pattern, payload signature based solutions [10] were developed; (2) assuming the existence of certain abnormal network activity or specific flow statistics, network anomaly detection based [6], flow feature [5] or communication pattern [4,11,12] detection based solutions were also developed; and (3) assuming the existence of similarities among bots, several behavior correlation based solutions have also been proposed [13,14]. In general, the more assumptions we make on botnets, the more restriction the corresponding botnet detection solutions suffer from, and accordingly, the easier being bypassed by modern botnets.

No matter how a botnet may change its behavior or appear differently, the motivation of botnets stays the same, which is to conduct certain profitable activities in underground market. Among all changeable appearances (e.g., spreading and control methods) of botnets, we classify them into three levels:

- Polymorphic appearance: Some features or characteristics of botnets are highly variable as designed. For example, the malware signatures. Various new system and network vulnerabilities will keep being discovered and exploited. The large number of software vendors and service providers will continue to contribute to this trend.
- Changeable appearance: C&C provides the channel between a botmaster and bots. Once certain C&C mechanism being well-studied and effective detection methods becoming available, new C&C mechanisms will show up soon. However, this type of changing is not as fast as we can find in polymorphic appearance of botnets. On the other hand, common infection and spreading methods may be more effective. Thus, for a certain time period, C&C can be still regarded as reliably recognizable appearance.
- Stable appearance: the final goal of a botnet is to perform certain profitable attack activities, such as spamming and DDoS attack. Even such behavior has become well-understood, botnets will not change it. New profitable activities may be discovered later. However, comparing to another two types of appearance, Stable Appearance becomes the directly recognizable one.

In this paper, we advocate a rather different approach called SeeBot to show another niche in detecting modern botnets. SeeBot is built upon a new active evidential reasoning model, which integrates the advantage of both passive and active monitoring and detection into one framework. In this framework, SeeBot focuses on recognizing stable appearances, which are related to Infection and Attack actions (I&A), and Command and Control activities (C&C), as its initial detection targets (i.e., passive reasoning). In our approach, if the passive evidential reasoning is not sufficient to detect botnets, SeeBot automatically selects optimal verification actions to discover relevant symptoms that are important to collect the most critical evidences.

Our contribution in this work is twofold:

- We propose an active multi-layer causality model to seamlessly integrate active detection actions into passive evidential reasoning process, such that the robustness and resilience of a botnet detection system can be significantly increased, especially when initial symptoms are weak.
- We design an open and incremental evidential reasoning framework to be adaptive and extensible to a variety of different monitoring sources, such that the applicability and practicability of the system can be greatly improved, especially for detecting new botnets.

The rest of the paper is organized as the following. Section 2 discusses related work on botnet detection. Section 3 proposes an active evidential reasoning based botnet detection model. Section 4 evaluates SeeBot performance using simulations and controlled experiments with real traffic traces, and the conclusion is given in Section 5.

2 Related work

Sustantial research work has been done in botnet detection. In the following, we briefly discuss several related work.

Extensive studies have been conducted on understanding the characteristics and behaviors of various botnets. To collect and analyze bots, researchers widely utilize honeypot techniques [15-17]. Freiling et al. [16] used honeypots to track botnets in order to explore a root-cause methodology to prevent DoS attacks. Nepenthes [15] is a special honeypot tool for automatic malware sample collection. Rajab et al. [17] provided an in-depth measurement study of the current botnet activities by conducting a longitudinal multi-faceted approach to collect bots and track botnets. Cooke et al. [18] conducted several

basic studies of botnet dynamics. In [19], Dagon et al. proposed to use DNS sinkholing technique for botnet study and pointed out the global diurnal behavior of botnets. Barford and Yegneswaran [20] provided a detailed study on the code base of several common bot families. Collins et al. [21] presented their observation of a relationship between botnets and scanning/spamming activities.

Several recent papers proposed different approaches to detect botnets. Ramachandran et al. [3] proposed using DNSBL (DNS blacklist) counter-intelligence to find botnet members that generate spams. This approach is useful for specific types of spam botnets. In [22], Reiter and Yen proposed a system TAMD to detect malware (including botnets) by aggregating traffic that shares the same external destination, similar payload, and that involves internal hosts with similar OS platforms. The corresponding aggregation method based on destination networks focuses on networks that experience an increase in traffic as compared to a historical baseline. Different from [14] that focuses on botnet detection. The scheme proposed in [22] aims to detect a broader range of malware.

Livadas et al. [4,11] proposed a machine learning based approach for botnet detection using some general network-level traffic features of chat-like protocols such as IRC. Karasaridis et al. [5] studied network flow level detection of IRC botnet controllers for backbone networks by matching a known IRC traffic profile. Rishi [10] is a signature-based IRC botnet detection system by matching known IRC bot nickname patterns. Binkley and Singh [6] proposed combining IRC statistics and TCP work weight for the detection of IRC-based botnets. Gu et al., 2007 [23] described BotHunter, which is a passive bot detection system that uses dialog correlation to associate IDS events to a user-defined bot infection dialog model. Different from BotHunter's dialog correlation or vertical correlation that mainly examines the behavior history associated with each distinct host, BotMiner utilizes a horizontal correlation approach that examines correlation across multiple hosts. BotSniffer [12] is an anomaly-based botnet C&C detection system that also utilizes horizontal correlation. However, it is used mainly for detecting centralized C&C activities (e.g., IRC and HTTP).

Many botnet detection solutions were designed based on certain assumptions on botnets with specific directly observable evidence (e.g., IRC botnet detection) or indirectly derivable evidence (e.g., correlation based botnet detection, a pattern of sequential observable network activities). Behavior similarity correlation based approach [13,14] cannot adapt to modern multi-function botnets with polymorphic behaviors.

More recently, p2p has been exploited as a new C&C mechanism in many modern botnets, which brings to a botnet detection system new challenges mainly on two aspects: (1) how to detect p2p traffic from background traffic; (2) how to distinguish C&C p2p traffic from legitimate p2p applications. Several solutions [24,25] have been proposed. Yen and Reiter, 2010 [24] showed that the different goals and circumstances, the features related to traffic volume, "churn" among peers, and differences between human-driven and machine-driven traffic make distinguishable behaviors in these p2p applications. Zhang et al., 2011 [25] proposed a novel botnet detection system based on statistical fingerprints to profile P2P traffic and identify stealthy P2P botnets.

In summary, we categorize those existing solutions designed based on certain pre-conditions (e.g., C&C mechanisms, sequential activities, behavior correlations) as condition-constrained botnet detection approach. All the solutions discussed here are in this category. To the best of our knowledge, SeeBot is the first active evidential reasoning framework for botnet detection that only assumes the intrinsic botnet activities. Many proposed solutions can work perfectly on detecting certain type of botnets as long as the expected pre-conditions are valid. The advantage of SeeBot lies in its high robustness, adaptability and applicability.

3 Active evidential reasoning

As discussed in Section 1, the three characteristics in modern botnet, namely highly polymorphic, intelligent and stealthy, make current botnet detection approaches miscellaneous and manifold. Botnet detection is essentially a process of detecting exposed botnet activities based on collected evidence.

An evidential reasoning approach usually uses a belief structure to model an assessment with uncertainty. In this section, we will formalize botnet detection as an evidential reasoning process. Accordingly, we will propose a new belief structure called Active Multi-layer Casuality Graph as shown in Figure 1, which seamlessly incorporate multiple components with different roles and levels, including Botnet, Evidence, Symptom and Actions, into the same active evidential reasoning framework.

In the following, we first introduce Active Multi-layer Casuality Graph, which is developed up the concept of casuality graph commonly used in evidential reasoning. Then we present the active evidential reasoning framework called SeeBot, and elaborate its functional modules.

3.1 Causality graph

A casualty graph [26] is a bipartite directed acylic graph to describe the Symptom-Cause correlation, which represents the causal relationship between each cause c_i and a set of observable symptoms S_{c_i} that may be triggered by c_i. Symptom-Cause causality graph provides a vector of correlation likelihood measure called likelihood indicator $I(s_j|c_i)$, to bind a root cause c_i to its relevant observable

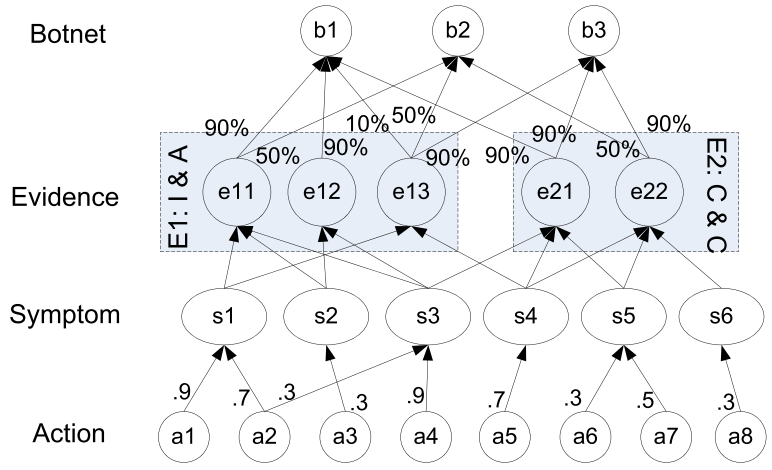

Figure 1 Active multi-layer casuality graph.

symptoms S_{c_i}. In a causality graph between root causes C and symptoms S, if $I(s_j|c_i) = 0$ or 1 for all (i,j), we call such causality model a deterministic model; otherwise, we call it a likelihood model.

However, a general casualty graph cannot satisfy the requirements in botnet detection, which can be mainly shown in the following two aspects.

First, the flat symptom structure in a casualty graph cannot completely represent the complicated relations among symptoms. A symptom may be the result from several other observed symptoms. For example, spamming, a symptom of common botnet attack, can be observed as the symptoms like (a) high volume of outbound TCP traffic with destination port TCP/25, and (b) multiple queries on different DNS MX records, etc. Accordingly, we break the original flat symptom structure into a two layer hierarchy between Symptom and Evidence to represent such complexity. Thus, a indirectly unobservable evidence can be jointly manifested by several directly observable symptoms. On the other hand, a symptom may contribute to believe the existence of different evidence.

Second, a general casualty graph can only represent a passive reasoning process. If any evidence is missing due to various reasons (e.g., packet loss), the passive reasoning result is commonly not satisfiable. Accordingly, in addition to the change made in the first step, we extend an action layer that is associated directly with the symptom layer to meet such requirement, which can selectively take actions to verify the most likely existed but lost evidence.

3.2 Active multi-layer causality graph

Active Multi-layer Causality Graph, denoted as AMCG and shown in Figure 1, consists of three bipartite directed acylic graphs hierarchically connected by different relationships. We use $B = \{b_1, b_2, \cdots, b_n\}$ to denote the cause set representing different types of botnet (e.g., IRC or P2P botnet), $E = \{e_1, e_2, \cdots, e_m\}$ to denote the evidence set that can be jointly used to detect the occurrence of botnet (e.g., P2P traffic, spamming), $S = \{s_1, s_2, \cdots, s_k\}$ to denote the symptom set that can be directly observed, and used to determine the existence of evidences (e.g., high Max degree ratio [27], high volume of SMTP traffic), and $A = \{a_1, a_2, \cdots, a_q\}$ to denote the action set used to check the symptoms.

There are two casualty correlations between B and E denoted as $M_{B\times E}$, as well as between E and S denoted as $M_{E\times S}$. $M_{B\times E}$ is used to define causal certainty between various botnet b_i ($b_i \in B$) and evidence e_j ($e_i \in E$). $M_{E\times O}$ is used to define causal certainty between evidence e_j ($e_i \in E$) and symptom s_k ($s_k \in S$). Evidence-Botnet causality graph provides a vector of correlation likelihood measure denoted as indication measure $I(e_j|b_i)$ to bind a type of botnet b_i to a set of its evidences E_{b_i}. Similarly, Symptom-Evidence causality graph provides a vector of correlation likelihood measure denoted as indication measure $I(s_k|e_j)$ to bind an evidence e_j to a set of its symptom S_{e_j}.

We also use $A = \{a_1, a_2, \cdots, a_q\}$ to denote the list of actions that can be used to check symptoms. We describe the relation between actions and symptoms using Action Book represented as a bipartite graph as shown in Figure 1. For example, the symptom s_1 can be verified using action a_1 or a_2. The Action Book can be defined by network managers based on symptom type, the network topology, and the available symptom validation tools.

The active multi-layer hybrid causality graph Botnet-Evidence-Symptom-Action graph is viewed as a 6-tuple $(B; E; S; A; C_1; C_2; C_3)$, where botnet set B, evidence set E, symptom set S, and action set A are four independent vertex sets. Every correlation edge in C_1 connects a vertex in E and a vertex in B to indicate causality relationship

between evidences and botnets. Every correlation edge in C_2 connects a vertex in O and a vertex in E to indicate causality relationship between symptoms and evidences. Every correlation edge in C_3 connects a vertex in A and a vertex in S to indicate verifiable relationship between actions and symptoms, referred as the Action Book.

3.3 Active evidential reasoning framework

SeeBot consists of four modules as shown in Figure 2, which are Evidence Mining (EM), Evidential Reasoning (ER), Plausible Reasoning (PR) and Action Selection (AS) modules. Evidence Mining module processes received symptoms from a passive network monitoring system and generate the corresponding evidences for all hosts in a monitored network based on Symptom-Evidence casuality relationship specified in AMCG. Evidential Reasoning module passively analyzes evidences, shows botnet likelihood evaluation for all hosts identified in from Evidence Mining module, and dynamically constructs a plausible graph presenting a plausible relationship between those hosts and each botnet category. Plausible Reasoning module identifies the smallest set of botnets to explain observed evidences for all related hosts, verifies if the confidence level of the reasoning result is satisfactory.

If the current related evidence is strong enough to explain the botnet hypothesis, then the reasoning process terminates. Otherwise, a list of most likely missing symptoms that can increase confidence on the botnet hypothesis is sent to Action Selection module. Selected actions are conducted to determine which unobserved symptoms have actually occurred and accordingly adjust

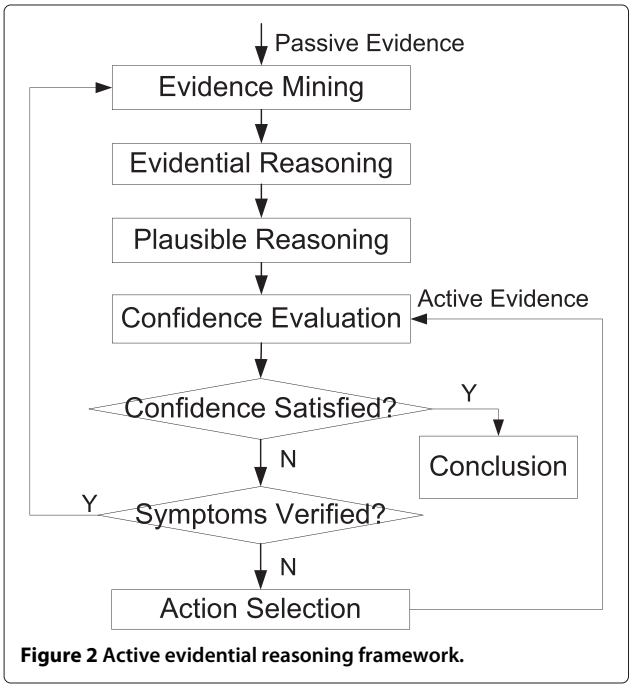

Figure 2 Active evidential reasoning framework.

hypothesis confidence level. If the new confidence level is satisfactory, then the reasoning process terminates; otherwise, the new symptom is fed into the fault reasoning module to create a new hypothesis. This process is recursively invoked until a highly credible hypothesis is found.

$$\begin{aligned} CF(b_i) &= CF_1(b_i) \times CF_2(b_i) \\ &= \prod_{t\in\{1,2\}} \left[1 - \prod_{e_j \in E_t} (1 - I(e_j) I(b_j|e_j)) \right] \end{aligned} \tag{1}$$

$$\begin{aligned} CF^{n+1}(b_i|e) &= \left[1-(1-CF_{T(e)})(1-I(e)I(b_i|e))\right] CF_{1-T(e)} \\ &= I(e)I(b_i|e)CF_{1-T(e)} + (1-I(e)I(b_i|e))CF^n \end{aligned} \tag{2}$$

3.3.1 *Evidence mining*

In this paper, we characterize a botnet using two essential coexisted behavior or evidence categories, namely (1) Infection and Attack (I&A) evidence denoted as E_1 that reflect botnet motivation, and (2) coordinated command and control (C&C) evidence denoted as E_2 that show the fundamental difference of botnet from other malwares. For detecting a botnet, we should observe evidences from E_1 *AND* E_2.

We use Symptom-Evidence bipartite casuality graph, as discussed in Section 3.1, to represent the complex relationships among various observations from different network and security monitoring systems. For *I*&*A* evidence category, we choose 10 E_1 evidences in the current version of SeeBot, denoted as $E_1 = \{e_{11}, \cdots, e_{1A}\}$, including scanning activity, spamming, DDoS, portable executable (PE) binary downloading, etc. For *C*&*C* evidence category, we choose 4 E_2 evidences in the current version of SeeBot, denoted as $E_2 = \{e_{21}, \cdots, e_{24}\}$, including P2P, IRC, HTTP, DNS. We use a parameter Evidence Degree (or *ED*) to indicate the number of associated symptoms. For example, as shown in Table 1, $ED(e_{11}) = 2$ and $ED(e_{21}) = 3$. More specifically, if we have observed $\{s_1, s_6, s_7\}$, we strongly believe there is P2P communication among certain investigated hosts [27]. Please note that Table 1 just shows an example of Evidence-Symptom-Action Casuality graph. Our current implementation of SeeBot has the average evidence degree 23 among all defined evidences.

In our model as the example shown in Table 1, multiple symptoms are required to verify the existence of evidence, denoted as $e_{11} \Leftarrow \{s_1, s_2\}$; on the other hand, one symptom may also be used as supportive observation for different evidence, denoted as $s_1 \rightarrow \{e_{11}, e_{21}\}$. Furthermore, our Symptom-Evidence causality adopts a deterministic model, which implies $I(s_k|e_j) = 1$ if e_j exists ($\forall s_k \in S_{e_j}$). We believe such simplification via the deterministic model is a necessary and effective step to release many typical botnet detection solutions from the burden in keeping

Table 1 An example of evidence-symptom-action casuality

Category	Evidence	Symptom	Action
E_1(*I&A*)	e_{11}(*Scanning*)	s_1(high TCP failure rate)	a_1 (snort)
		s_2(fast varying dest ports)	a_2 (argus)
	e_{12}(*Spamming*)	s_3(high volume TCP/25)	a_3 (NetFlow)
		s_4(multi DNS MX queries)	a_4 (DNS log)
		s_5(multi SMTP dest)	a_5 (snort)
E_2(*C&C*)	e_{21}(*P2P*)	s_1(high TCP failure rate)	a_1 (snort)
		s_6(high In-and-Out degree)	a_6 (Script 1)
		s_7(high Max Degree Ratio)	a_7 (Script 2)

track of the difference and details of various botnets. The input of this module is various tracking symptoms, and the output is all related evidences and their indication strength for each host relating to those symptoms.

3.3.2 Evidential reasoning

Evidence shows certain indication that leads to a decision with more or less uncertainty. The stronger indication an evidence shows, the less uncertainty a conclusion remains. Traditional uncertainty reasoning approaches based on Bayesian Network and Dempster-Shafer theory are inapplicable to intrusion detection due to lack of prior knowledge [28]. In our system, we adopt the scaling mechanism proposed in [28] to classify available evidence into three levels: strong (S), moderate (M), and weak (W) indications. Further, we evaluate a decision on botnet detection into three likelihood levels as well: strong, moderate and weak confidence. To facilitate the reasoning process and practical operation, we empirically quantify each evidence level with numerical values $S = 0.9, M = 0.5$ and $W = 0.1$ for strong, moderate, and weak likelihood levels.

In evidential reasoning, the confidence on botnet detection (CF), as shown in Eq. 1, depends on the confidence on the detection of I&A (CF_1) and C&C (CF_2). Furthermore, CF_1 and CF_2 rely on the joint events and their likelihood levels (i.e., S, M, W) from E_1 and E_2 respectively. In Eq. 1, $I(e_j) = \frac{|S^o_{e_j}|}{|S_{e_j}|}$, is denoted as evidence indication strength. If one type of evidence is completely missing, we use a null evidence (e_u) to represent, and $I(e_u) = 0$. In such a case, $CF = 0$.

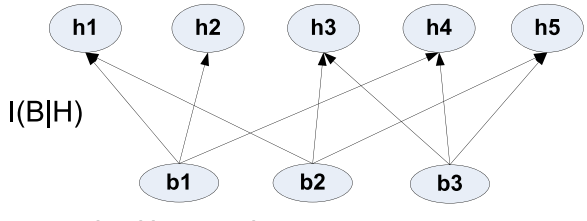

Figure 3 Plausible Bot graph.

Next, we present a method to incrementally update current confidence level denoted as CF^n to a new level CF^{n+1} when a new evidence e with $I(e) = x$ is observed. Here, we use a variable $T(e)$ ($T(e) \in \{0, 1\}$) to represent the event category. If $T(e) = 0$, $e \in E_1$; otherwise, $e \in E_2$. Since $CF^n = CF_1 \times CF_2 = (1 - (1 - CF_1))(1 - (1 - CF_2))$, we derive the method as shown in Eq. 2 to incrementally update current confidence.

3.3.3 Plausible reasoning

After the process of Belief Reasoning, a *Plausible Bot Graph* as shown in Figure 3 is constructed as the following to represent the dynamic relationship between the potential faulty components and the evidence.

- For each botnet in B, to associate a botnet vertex b_i.
- For each investigated host component, to associate a host vertex h_j. with its total belief metric Ψ_c.
- For each investigated host component h_j, to associate a link to all its related botnets with the weight $I(b_i|h_j)$ (here, $I(b_i|h_j) = CF(b_i)$).

A *plausible botnet reasoning* problem is to find the minimal number of most likely botnets based on *Plausible Bot Graph* that explain all related evidences with investigated hosts.

Theorem 1. *A plausible botnet reasoning problem is NP-complete.*

Proof. A plausible botnet reasoning problem can be reduced to a Weighted Set Cover (WSC) problem as follows. For a set of investigated hosts $H = \{h_1, h_2, \cdots, h_n\}$, a cover is defined as a subset of hosts H_{b_i} that can be explained by one botnet b_i. Each cover H_{b_i} is assigned a weight as $W(H_{b_i}) = 1/\sum_{h_j \in H_{b_i}} I(b_i|h_j)$. Obviously, the less weight a cover is, it is more likely that the corresponding botnet is the cause. Thus, the plausible botnet reasoning is to find a collection of covers (i.e., botnets) such that $\bigcup_{b_i \in B} H_{b_i} = H$ with $min(\sum_{b_i \in B} W(H_{b_i}))$. This is a Weighted Set Cover problem that is NP-complete. □

We adopt greedy heuristic algorithm [26] for the plausible botnet reasoning problem as shown in Algorithm 1, where B is a set of potentially botnets; H is a set of investigated hosts; R is a set of inferred botnets.

Algorithm 1 Plausible Botnet Reasoning Algorithm.

Step 1. $R \leftarrow \emptyset$;
Step 2. Find a botnet b_i ($b_i \in B$) with $min(W(H_{b_i}))$;
Step 3. $R \leftarrow R \cup \{b_i\}$;
Step 4. $H = H - H_{b_i}$;
Step 5. Go to Step 2 until $H = \emptyset$.

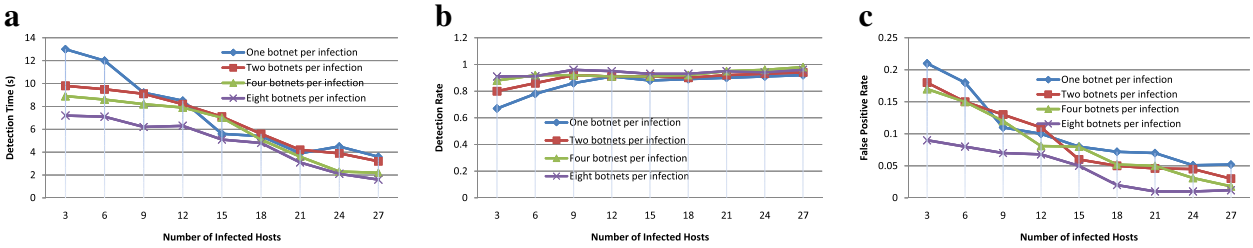

Figure 4 Performance evaluation. (a) Detection time (s) **(b)** Detection rate **(c)** False Positive Rate.

3.3.4 *Action selection*

Figure 1 presents the verification relationship between evidence $\{e_1, e_2, \cdots\}$ and actions $\{a_1, a_2, \cdots\}$. For example, Evidence e_1 can be verified by taking a combination of action a_1, a_2 and a_3, which can be denoted as a new virtual action vertex v_1 associated with a cost of the sum of $C(a_1), C(a_2)$ and $C(a_3)$. Action v_1 can verify all symptoms (s_1, s_2) that are verifiable by either a_1, a_2 or a_3. After converting joint actions to a virtual action, Symptom-Action correlation can be represented in a bipartite graph.

The goal of the Action Selection algorithm is to select the actions that cover all evidences E_{UO} with a minimal action cost. Based on Symptom-Action bipartite graph, we can model this problem as WSC problem to solve it [26].

4 Evaluation

In this section, we present our evaluation metrics, experiment methodology and experiment results.

4.1 Evaluation metrics

We evaluate SeeBot from the following three aspects: performance, accuracy and sensibility.

The performance of SeeBot is measured by botnet detection time τ, which is the time between receiving the first evidence (i.e., when malware becomes active) and identifying the actual botnets. The accuracy of SeeBot depends on two factors: (1) the detection ratio (μ), which is the ratio of the number of true detected botnets (B_d is the total detected fault set) to the number of actual occurred botnets B_h, formally $\mu = \frac{|B_d \cap B_h|}{B_h}$; and (2) false positive ratio (ν), which is the ratio of the number of false reported botnets to the total number of detected botnet, formally $\nu = \frac{|B_d - B_d \cap B_h|}{B_d}$. We have also analyzed system sensitivity showing the impact of the number of evidences and their evidence degrees on the system performance and accuracy.

4.2 Experiment methodology

One challenge in evaluating botnet detection solutions is the lack of group truth. The evaluation method adopted in our evaluation is called active evaluation, in which we execute real botnet binaries in a controlled network environment. By properly selecting different type of botnets, we can clearly demonstrate the applicability and robustness of SeeBot in detecting botnets.

Our controlled experiment environment has the following configuration: a Cisco ASA firewall with NetFlow enabled, a Cisco switch with SPAN port connecting a Snort IDS, and a event monitoring server with SeeBot installed for analyzing various symptoms, including Snort alerts, syslogs, DNS logs and host events (e.g., IRC activity log, HIDS events, AVG anti-virus application, malware alerts) from distributed systems. In our system, network

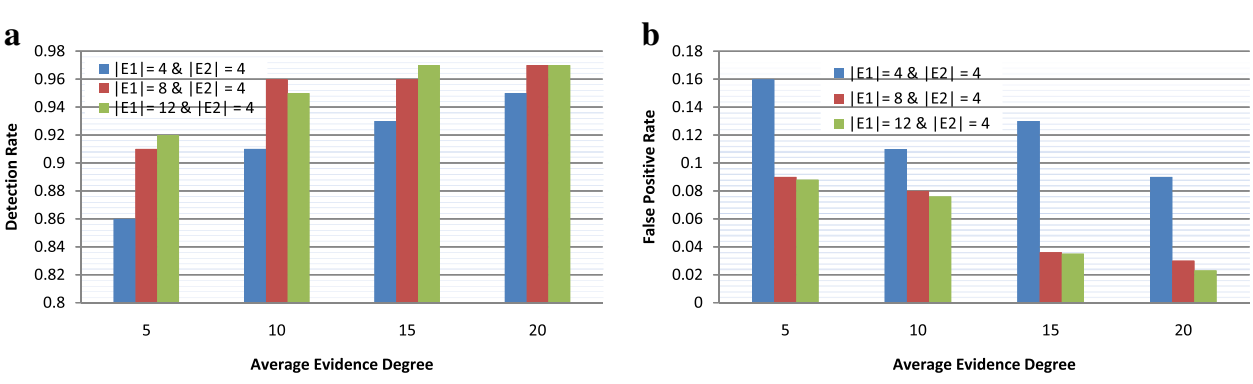

Figure 5 Evidence degree impact. (a) The impact of evidence degree on detection rate. **(b)** The impact of evidence degree on false positive rate.

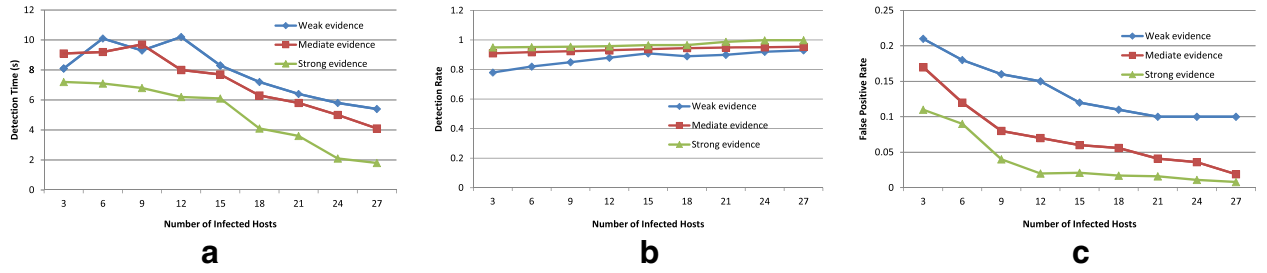

Figure 6 The impact of evidence indication strength on **(a)** Detection time (s), **(b)** Detection rate, and **(c)** False Positive Rate.

related evidence is adaptively combined with host related evidence. More specifically, network evidence will be initially analyzed at the first stage. The confidence evaluation on the generated detection hypotheses will be consistently conducted and incrementally updated with new evidence either passively input to actively collected from the monitored network system. Host based evidence such as security logs will be collected only when the passively input network related evidence is insufficient.

The testbed consists of eight physical workstations and 20 virtual machines, with Ubuntu and Windows XP installed. The system has been directly connected to th1e Internet for several months to record all traffic traces that will be used later as the background traffic when we disconnect the testbed from public network and run malicious software. More than 500 different type of botnet binaries have been collected over one year, and we selected 24 binaries among them, including 9 IRC botnet, 8 HTTP botnet, and 7 P2P botnets. In each test, the number of infected machines M increased from 3 to 27 at a speed 3 each time. For each test run, we randomly copied 1, 2, 4, 8 botnet binaries to the selected infection machines and execute them. In order to evaluate the impact of SeeBot parameters, we set up SeeBot with 3 different E_1/E_2 configurations as: 4/4, 8/4, and 12/4. For each event configuration (e.g., 4/4, standing for 4 E_1 and 4 E_2 events), the average event degrees (ED) are set to 5, 10, 15, and 20.

4.3 Experiment results

Figure 4 shows the performance evaluation results. Figure 4-(a) indicates SeeBot can detect botnets effectively (average in 5s), especially when more hosts infected. This is one obvious feature in an evidential reasoning system. Since SeeBot is not designed for any specific type of botnets, the evidence from multiple mixed botnets can jointly increase the detection performance, which is also clearly shown in Figure 4-(a). Figure 4-(b) shows the detection rate is above 90% on average with false positive rate less than 5% on average as shown in Figure 4-(c). In our framework, we identify the I&A and C&C related behaviors represent intrinsic botnet activities. If the evidence related to one of these two behaviors is significantly missing, SeeBot will not perform well initially. However, active reaction feature in SeeBot can effectively improve its detection rate by searching for missing evidence.

Figure 5 implies that with the increase of evidences and the corresponding associated symptoms, the overall performance of SeeBot is evidently increased. However, the improvement due to increasing evidences and symptoms becomes much slow after certain thresholds as the total evidences $|E_1| = 8$, $|E_2| = 4$ and the average evidence degree 15. Such an observation shows SeeBot can perform equivalently good as long as certain amount of evidence available to the system.

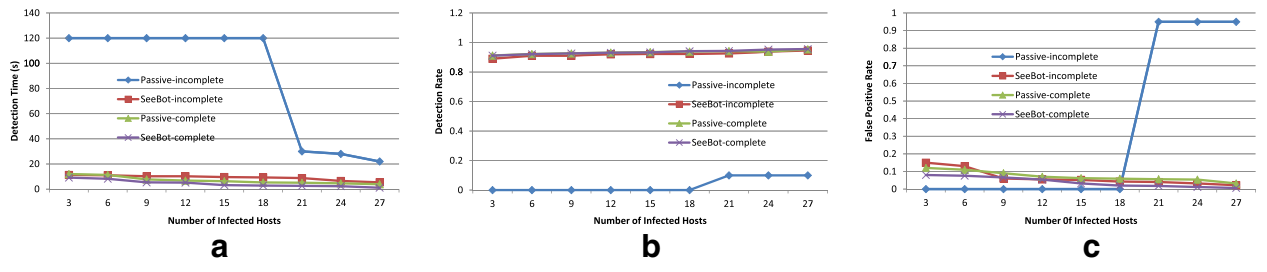

Figure 7 The performance comparison between SeeBot and the passive approach on **(a)** Detection time (s), **(b)** Detection rate, and **(c)** False Positive Rate.

As shown in Figure 5 and the following experiment results, the false positive rate sometime can be high. The main reason of causing false positive results is due to the multi-association (i.e., one symptom may associate to multiple evidence) between the elements in the evidence and symptom sets, especially when some symptoms are not detected or lost.

SeeBot provides an open framework such that different symptoms, evidences and botnets can be selected and associated together. Apparently, the evidence indication strength (as defined in Sec.3.3.2) between associated evidences and botnets can directly affect the system performance. Apparently, the general rule is to associate "Strong" evidences with botnets. However practically, the feasibility of constructing such a framework may depend on the configuration of a real network system. To clearly characterize the impact of the evidence indication strength on the performance of SeeBot, we have conducted experiments in three cases with all weak, all mediate and all strong evidences to validate the SeeBot performance. In our experiments, the different infection rates show the similar results. Thus, we only show the experiment results with two botnets per infection. As shown in Figure 6, the different evidence indication strength does cause some differences (10%–20% difference) on both detection time and detection rate, and makes significant difference on false positive rate (50%–500% difference). The time difference is caused by the consumed time in active reasoning process when SeeBot tries to increase the confidence level on the detection results. Even with weak evidence, SeeBot can still localize relevant botnets the same way as its receiving strong evidence. Thus, the difference on detection rate is not obvious. Since weak evidence is typically associated suspicious botnets, the false positive rate can be really high due to such ambiguity.

Many existing botnet detection tools were designed based on various assumptions, e.g., the existence of dialog model in BotHunter. The advantage of SeeBot lies in its reasoning capability in botnet detection, especially when available evidence is incomplete, which does not rely on certain botnet behavior to work. Since various botnets are used in our evaluation, for a fair comparison to the related work, we have implemented the essential passive reasoning system based on the proposed solution from [2,23,27]. To illustrate the difference in the passive approach and SeeBot, we further created two scenarios: (1) with incomplete evidence; (2) with complete evidence. As shown in Figure 7, when the observable evidence is complete, the difference between the two approaches are not obvious. However, when we only provided incomplete evidence to the two systems, the advantage of SeeBot over the passive approach is very significant. For instance, the passive approach cannot even start working (we set two minutes as a timeout threshold) when the evidence availability (i.e., 80% of evidence missing in our experiment) is low. Even when evidence availability rate increase to 50% such that the passive system started working, SeeBot is clearly superior to the passive approach as shown in Figure 7.

5 Conclusion and future work

In this paper, we advocate a rather different approach called SeeBot to show another niche in detecting modern botnets. SeeBot is built upon a new active evidential reasoning model, which integrates the advantage of both passive and active monitoring and detection into one framework. The experiments show that SeeBot can effectively detect modern botnets, especially when the initial evidence is not evident. Our future work includes the development of an uncertainty evaluation model to provide a confidence measurement on the detection results. We also feel it is important that SeeBot can detect hidden common characteristics embedded in collected evidences to characterize new botnets. The future version of SeeBot could be enhanced with certain effective learning mechanism such as reinforcement learning to achieve such a desirable feature.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
YT et al. proposed an active multi-layer causality model to seamlessly integrate active detection actions into passive evidential reasoning process. They designed an open and incremental evidential reasoning framework to be adaptive and extensible to a variety of different monitoring sources. All authors read and approved the final manuscript.

Author details
[1]School of Information Technology, Illinois State University, Normal, IL 61790, USA. [2]School of Computer Science and Engineering, Southeast University, Nanjing, P.R. China. [3]National Computer Network Key Laboratory, Southeast University, Nanjing, P.R. China. [4]School of Computing, DePaul University, Chicago, IL 60604, USA.

Received: 14 November 2013 Accepted: 29 November 2013
Published: 13 December 2013

References

1. Microsoft's SIRv9 (2010) Security Intelligence Report volume 9. http://www.microsoft.com/security/sir
2. Shin S, Lin R, Gu G (2011) Cross-analysis of Botnet victims: new insights and implications. In: Proceedings of the 14th International Symposium on Recent Advances in Intrusion Detection (RAID 2011), Menlo Park, California, September 2011
3. Ramachandran A, Feamster N, Dagon D (2006) Revealing botnet membership using DNSBL counterintelligence. In: Proceedings of USENIX SRUTI'06
4. Strayer WT, Walsh R, Livadas C, Lapsley D (2006) Detecting botnets with tight command and control. In: Proceedings of the 31st IEEE Conference on Local Computer Networks (LCN'06)
5. Karasaridis A, Rexroad B, Hoeflin D (2007) Widescale botnet detection and characterization. In: Proceedings of USENIX HotBots'07
6. Binkley JR, Singh S (2006) An algorithm for anomaly-based botnet detection. In: Proceedings of USENIX SRUTI'06, July 2006, pp 43–48

7. Kartaltepe EJ, Morales JA, Xu S, Sandhu R (2010) Social network-based Botnet command-and-control: emerging threats and countermeasures. Applied Cryptography and Network Security. Lecture Notes in Computer Science Volume 6123, pp 511–528
8. Damballa Top 10 Botnet Threat Report 2010. http://www.damballa.com/downloads/r_pubs/Damballa_2010_Top_10_Botnets_Report.pdf
9. Zou CC, Cunningham R (2006) Honeypot-Aware Advanced Botnet Construction and Maintenance. In: the International Conference on Dependable Systems and Networks (DSN). June 25-28, p.199-208, Philadelphia
10. Goebel J, Holz T (2007) Rishi: Identify bot contaminated hosts by irc nickname evaluation. In: Proceedings of USENIX HotBots'07
11. Livadas C, Walsh R, Lapsley D, Strayer WT (2006) Using machine learning techniques to identify botnet traffic. In: Proceedings of the 2nd IEEE LCN Workshop on Network Security (WoNS'2006)
12. Gu G, Zhang J, Lee W (2008) BotSniffer: Detecting botnet command and control channels in network traffic. In: Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)
13. Zeng Y, Hu X, Shin KG (2010) Detection of Botnets Using Combined Host- and Network-Level Information. In: Proceedings of the 40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2010), Chicago
14. Gu G, Perdisci R, Zhang J, Lee W (2008) BotMiner: clustering analysis of network traffic for protocol- and structure-independent botnet detection. In: Proceedings of the 17th USENIX Security Symposium (Security'08), San Jose, CA
15. Baecher P, Koetter M, Holz T, Dornseif M, Freiling F (2006) The nepenthes platform: an efficient approach to collect malware. In: Proceedings of International Symposium on Recent Advances in Intrusion Detection (RAID'06), Hamburg, September 2006
16. Freiling F, Holz T, Wicherski G (2005) Botnet tracking: exploring a root-cause methodology to prevent denial of service attacks. In: Proceedings of 10th European Symposium on Research in Computer Security (ESORICS'05)
17. Rajab M, Zarfoss J, Monrose F Terzis A (2006) A multi-faceted approach to understanding the botnet phenomenon. In: Proceedings of ACM SIGCOMM/USENIX Internet Measurement Conference (IMC'06), Brazil, October 2006
18. Cooke E, Jahanian F, McPherson D (2005) The zombie roundup: understanding, detecting, and disrupting botnets. In: Proceedings of USENIX SRUTI'05
19. Dagon D, Zou C, Lee W (2006) Modeling botnet propagation using timezones. In: Proceedings of the 13th Annual Network and Distributed System Security Symposium (NDSS'06), January 2006
20. Barford P, Yegneswaran V (2006) An inside look at Botnets. In: Special Workshop on Malware Detection, Advances in Information Security. Springer Verlag
21. Collins M, Shimeall T, Faber S, Janies J, Weaver R, Shon MD, Kadane J (2007) Using uncleanliness to predict future botnet addresses. In: Proceedings of ACM/USENIX Internet Measurement Conference (IMC'07)
22. Reiter MK, Yen T-F (2008) Traffic aggregation for malware detection. In: Proceedings of the Fifth GI International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA'08)
23. Gu G, Porras P, Yegneswaran V, Fong M, Lee W (2007) Bothunter: detecting malware infection through ids-driven dialog correlation. In: 16th USENIX Security Symposium (Security'07)
24. Yen T-F, Reiter MK (2010) Are your hosts trading or plotting? Telling P2P file-sharing and Bots apart. In: The 2010 IEEE 30th International Conference on Distributed Computing Systems (ICDCS)
25. Zhang J, Perdisci R, Lee W, Sarfraz U, Luo X (2011) Detecting stealthy P2P Botnets using statistical traffic fingerprints. In: The 41th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN2011), Hong Kong, China
26. Tang Y, Al-Shaer E, Boutaba R (2008) Efficient fault diagnosis using incremental alarm correlation and active investigation for internet and overlay networks. IEEE Trans Netw Serv Manage 5(1):36–49
27. Iliofotou M, Pappu P, Faloutsos M, Mitzenmacher M, Varghese G, Kim H (2008) Graption: Automated detection of P2P applications using traffic dispersion graphs (TDGs). In: UC Riverside Technical Report, CS-2008-06080
28. Ou X, Raj Rajagopalan S, Sakthivelmurugan S (2009) An empirical approach to modeling uncertainty in intrusion analysis. In: Annual Computer Security Applications Conference (ACSAC), Honolulu, Hawaii, USA, Dec 2009

doi:10.1186/1869-0238-4-20
Cite this article as: Tang *et al.*: **Catching modern botnets using active integrated evidential reasoning.** *Journal of Internet Services and Applications* 2013 **4**:20.