Detecting Intentional Packet Drops on the Internet via TCP/IP Side Channels

Roya Ensafi, Jeffrey Knockel, Geoffrey Alexander, and Jedidiah R. Crandall

Department of Computer Science, University of New Mexico, USA
{royaen,jeffk,alexandg,crandall}@cs.unm.edu

Abstract. We describe a method for remotely detecting intentional packet drops on the Internet *via* side channel inferences. That is, given two arbitrary IP addresses on the Internet that meet some simple requirements, our proposed technique can discover packet drops (*e.g.*, due to censorship) between the two remote machines, as well as infer in which direction the packet drops are occurring. The only major requirements for our approach are a client with a global IP Identifier (IPID) and a target server with an open port. We require no special access to the client or server. Our method is robust to noise because we apply intervention analysis based on an autoregressive-moving-average (ARMA) model. In a measurement study using our method featuring clients from multiple continents, we observed that, of all measured client connections to Tor directory servers that were censored, 98% of those were from China, and only 0.63% of measured client connections from China to Tor directory servers were not censored. This is congruent with current understandings about global Internet censorship, leading us to conclude that our method is effective.

1 Introduction

Tools for discovering intentional packet drops are important for a variety of applications, such as discovering the blocking of Tor by ISPs or nation states [1]. However, existing tools have a severe limitation: they can only measure when packets are dropped in between the measurement machine and an arbitrary remote host. The research question we address in this paper is: can we detect drops between two hosts without controlling either of them and without sharing the path between them? Effectively, by using idle scans our method can turn approximately 1% of the total IP address space into conscripted measurement machines that can be used as vantage points to measure IP-address-based censorship, without actually gaining access to those machines.

Antirez [2] proposed the first type of idle scan, which we call an IPID idle port scan. In this type of idle scan an "attacker" (which we will refer to as the *measurement machine* in our work) aims to determine if a specific port is open or closed on a "victim" machine (which we will refer to as the *server*) without using the attacker's own return IP address. The attacker finds a "zombie" (*client* in this paper) that has a global IP identifier (IPID) and is completely idle. In this

M. Faloutsos and A. Kuzmanovic (Eds.): PAM 2014, LNCS 8362, pp. 109–118, 2014.

paper, we say that a machine has a global IPID when it sends TCP RST packets with a globally incrementing IPID that is shared by all destination hosts. This is in contrast to machines that use randomized IPIDs or IPIDs that increment per-host. The attacker repeatedly sends TCP SYN packets to the victim using the return IP address of the zombie, while simultaneously eliciting RST packets from the zombie by sending the zombie SYN/ACKs with the attacker's own return IP address. If the victim port that SYN packets are being sent to is open, the attacker will observe many skips in the IPIDs from the zombie. Nmap [3] has built-in support for the IPID idle scan, but often fails for Internet hosts because of noise in the IPID that is due to the zombie sending packets to other hosts. Our method described in this paper is resistant to noise, and can discover packet drops in either direction (and determine which direction). Nmap cannot detect the case of packets being dropped from client to server based on destination IP address, which our results demonstrate is a very important case.

Two other types of idle scans were presented by Ensafi *et al.* [4], including one that exploits the state of the SYN backlog as a side channel. Our method is based on a new idle scan technique that can be viewed as a hybrid of the IPID idle scan and Ensafi *et al.*'s SYN backlog idle scan. Whereas Ensafi *et al.*'s SYN backlog idle scan required filling the SYN backlog and therefore causing denial-of-service, our technique uses a low packet rate that does not fill the SYN backlog and is non-intrusive. The basic insight that makes this possible is that information about the server's SYN backlog state is entangled with information about the client's IPID field. Thus, we can perform both types of idle scans (IPID and SYN backlog) to detect drops in both directions, and our technique overcomes the limitations of both by exploiting the entanglement of information in the IPID and treating it as a linear intervention problem to handle noise characteristic of the real Internet.

This research has several major contributions:

- A non-intrusive method for detecting intentional packet drops between two IP addresses on the Internet where neither is a measurement machine.
- An Internet measurement study that shows the efficacy of the method.
- A model of IPID noise based on an autoregressive-moving-average (ARMA) model that is robust to autocorrelated noise.

Source code and data are available upon request, and a web demonstration version of the hybrid idle scan is at `http://spookyscan.cs.unm.edu`. The types of measurements we describe in this paper raise ethical concerns because the measurements can cause the appearance of connection attempts between arbitrary clients and servers. In China there is no evidence of the owners of Internet hosts being persecuted for attempts to connect to the Tor network, thus our measurements in this paper are safe. However, we caution against performing similar measurements in other countries or contexts without first evaluating the risks and ethical issues. More discussion of ethical issues, additional details about the ARMA model, and other information not included here due to space limitations are available in the extended version of this paper [5].

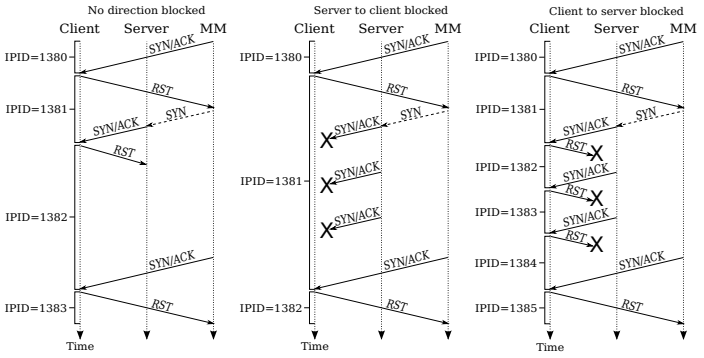

Fig. 1. Three different cases that our method can detect. MM is the measurement machine.

The rest of the paper is structured as follows: After describing the implementation of our method in Section 2, we present our experimental methodology for the measurement study in Section 3 and the ARMA model in Section 4. Results from the measurement study are in Section 5, followed by a discussion of related work in Section 6 and then the conclusion.

2 Implementation

In order to determine the direction in which packets are being blocked, our method is based on information flow through both the IPID of the client and the SYN backlog state of the server, as shown in Figure 1. Our implementation queries the IPID of the client (by sending SYN/ACKs from the measurement machine and receiving RST responses) to create a time series to compare a base case to a period of time when the server is sending SYN/ACKs to the client (because of our forged SYNs). We assume that the client has global IPIDs and the server has an open port.

Global IPIDs were explained in Section 1. The SYN backlog is a buffer that stores information about half-open connections where a SYN has been received and a SYN/ACK sent but no ACK reply to the SYN/ACK has been received. Half-open connections remain in the SYN backlog until the connection is completed with an ACK, aborted by a RST or ICMP error, or the half-open connection times out (typically between 30 and 180 seconds). The SYN/ACK is retransmitted some fixed number of times that varies by operating system and version, typically three to six SYN/ACKs in total. This SYN backlog behavior on the server, when combined with the global IPID behavior of the client, enables us to distinguish three different cases (plus an error case):

– **Server-to-client-dropped:** In this case SYN/ACKs are dropped in transit from the server to the client based on the return IP address (and possibly

other fields like source port), and the client's IPID will not increase at all (except for noise).

- **No-packets-dropped:** In the case that no intentional dropping of packets is occurring, the client's IPID will go up by exactly one. This happens because the first SYN/ACK from the server is responded to with a RST from the client, causing the server to remove the entry from its SYN backlog and not retransmit the SYN/ACK. Censorship that is stateful or not based solely on IP addresses and TCP port numbers may be detected as this case, including filtering aimed at SYN packets only. Also, if the packet is not dropped, but instead the censorship is based on injecting RSTs or ICMP errors, it will be detected as this case. Techniques for distinguishing these other possibilities are left for future work.
- **Client-to-server-dropped:** In this case RST responses from the client to the server are dropped in transit because of their destination IP address (which is the server). When this happens the server will continue to retransmit SYN/ACKs and the client's IPID will go up by the total number of transmitted SYN/ACKs including retransmissions (typically three to six). This may indicate the simplest method for blacklisting an IP address: null routing.
- **Error:** In this case networking errors occur during the experiment, the IPID is found to not be global throughout the experiment, a model is fit to the data but does not match any of the three non-error cases above, the data is too noisy and intervention analysis fails because we are not able to fit a model to the data, and/or other errors.

Noise due to packet loss and delay or the client's communications with other machines may be autocorrelated. The autocorrelation comes from the fact that the sources of noise, which include traffic from a client that is not idle, packet loss, packet reordering, and packet delay, are not memoryless processes and often happen in spurts. The accepted method for performing linear intervention analysis on time series data with autocorrelated noise is ARMA modeling, which we describe in Section 4.

3 Experimental Setup

All measurement machines were Linux machines connected to a research network with no packet filtering. Specifically, this network has no stateful firewall or egress filtering for return IP addresses.

One measurement machine was dedicated to developing a pool of both client and server IP addresses that have the right properties for use in measurements. Clients were chosen by horizontally scanning China and other countries for machines with global IPIDs, then continually checking them for a 24-hour period to cull out IP addresses that frequently changed global IPID behavior (*e.g.*, because of DHCP), went down, or were too noisy. A machine is considered to have a global IPID if its IPID as we measure it by sending SYN/ACKs from alternating

source IP addresses and receiving RSTs never incrementing outside the ranges $[-40, 0)$ or $(0, 1000]$ per second when probed from two different IP addresses. This range allows for non-idle clients, packet loss, and packet reordering. It is possible to build the time series in different ways where negative IPID differences are never observed, but in this study our time series was the differences in the client's IPIDs in the order in which they arrived at the measurement machine. Our range of $[-40, 0)$ or $(0, 1000]$ is based on our observations of noise typical of the real Internet. The IPID going up by 0 is a degenerate case and means the IPID is not global.

Servers were chosen from three groups: Tor directory authorities, Tor bridges, and web servers. The ten Tor directory authorities were obtained from the Tor source code and the same ten IPs were tested for every day of data. Three Tor bridges were collected daily both through email and the web. Every day seven web servers were chosen randomly from the top 1000 websites on the Alexa Top 1,000,000 list [6]. All web server IPs were checked to make sure that they stood up for at least 24 hours before being selected for measurement. Furthermore, we checked that the client and server were both up and behaving as assumed between every experiment (*i.e.*, every five minutes).

A round of experiments was a 24-hour process in which measurements were carried out on the two measurement machines. Each 24-hour period had 22 hours of experiments and 2 hours of down time for data synchronization. For each measurement period on each of the two machines performing direct measurements, ten server machines and ten client machines from the above process were chosen for geographic diversity: 5 from China, 2 from countries in Asia that were not China, 1 from Europe, and 2 from North America. IP addresses were never reused except for the Tor directory authorities, so that every 24-hour period was testing a set of 20 new clients, 10 new servers, and the 10 directory authorities.

For each of the twenty clients and twenty servers geographical information provided by MaxMind was saved. MaxMind claims an accuracy of 99.8% for identifying the country an IP address is in [7]. For each of the twenty server machines, a series of SYN packets was used to test and save its SYN/ACK retransmission behavior for the analysis in Section 4.

Every hour, each of our two main measurement machines created ten threads. Each thread corresponded to one client machine. Each thread tested each of the ten server IP addresses sequentially using our idle scan based on the client's IPID. No forged SYNs were sent to the server during the first 100 seconds of a test, and forged SYNs with the return IP address of the client were sent to the server at a rate of 5 per second for the second 100-second period. Then forged RST packets were sent to the server to clear the SYN backlog and prevent interference between sequential experiments. A timeout period of sixty seconds was observed before the next test in the sequence was started, to allow all other state to be cleared. Each experiment lasted for less than five minutes, so that ten could be completed in an hour. Every client and server was involved in only one experiment at a time. Each client/server pair was tested once per hour throughout the 24-hour period, for replication and also to minimize the effects

of diurnal patterns. Source and destination ports for all packets were carefully chosen and matched to minimize assumptions about what destination ports the client responds on.

4 Analysis

In this section we give an overview of our intervention analysis based on ARMA modeling. More details are available in the extended version of the paper [5].

We model each time series $y_1, \ldots, y_n$ as a *linear regression with ARMA errors*, a combination of an autoregressive-moving-average (ARMA) model with external linear regressors. An ARMA(p, q) model combines an AR model of order p and an MA model of order q. We use a linear regression with ARMA errors to model our time series data. This specifies that every element in a time series can be written as a constant plus the linear combination of regressors $x_1, \ldots, x_r$ with an ARMA-modeled error term, e_t:

$$y_t = c + \sum_{i=1}^{r} \beta_i x_{it} + e_t, \qquad e_t = z_t + \sum_{i=1}^{p} \phi_i e_{t-i} + \sum_{i=1}^{q} \theta_i z_{t-i}$$

where z_t is a white noise series and ϕ_i, θ_i, and β_i are ARMA model parameters to be fitted. We use the regressors x_i for *intervention analysis*, *i.e.*, for analyzing our experimental effect on the time series at a specific time.

For each experiment, we pick regressors according to which times the server (re)transmits SYN/ACK's in response to SYN's. For a server that (re)transmits r SYN/ACK's in response to each SYN, we have r regressors. We call time t_1 the time of the first transmission in response to the first of our forged SYN's, and we call t_{i+1} the time the server would send the ith retransmission in response to that SYN. Then we define regressor x_i as the indicator variable

$$x_{ij} = \begin{cases} 1 & \text{if } t_i \leq j \text{ and either } j < t_{i+1} \text{ or } i = r \\ 0 & \text{otherwise} \end{cases}$$

In other words, x_1 is zeros until the time the server transmits the first SYN/ACK then ones until the server begins retransmitting SYN/ACK's. The remaining x_i are zeros until the time the server would begin retransmitting its ith SYN/ACK then ones until if/when the $(i+1)$th SYN/ACK's would begin being retransmitted. This definition allows us to model any of the possible level shifts in any case of packet drop as a linear combination of all x_i. See Figure 2 for an illustration.

For intervention analysis, we use hypothesis testing over a value β_r, which represents the difference in IPID differences between when we do or do not send forged SYN packets to the server. Then we determine the case by a series of one-sided hypothesis tests performed with significance $\alpha = 0.01$ according to the following breakdown, where k_1 and k_2' are thresholds between cases:

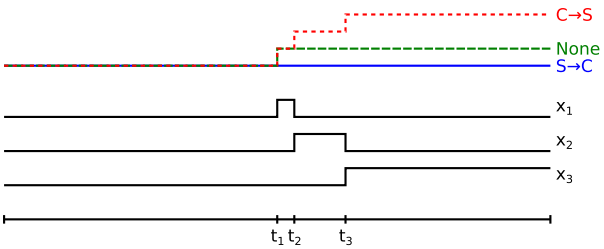

Fig. 2. For a server that retransmits $r-1$ SYN/ACK's, each case can be expressed as the linear combination of regressors $x_1, \ldots, x_r$; shown is when $r = 3$ with SYN/ACK transmissions responding to the first forged SYN occurring at t_1, t_2, and t_3

- **Server-to-client-dropped** if we reject the null hypothesis that $\beta_r \geq k_1$.
- **No-packets-dropped** if we reject the null hypotheses that $\beta_r \leq k_1$ and that $\beta_r \geq k_2'$.
- **Client-to-server-dropped** if we reject the null hypothesis that $\beta_r \leq k_2'$.
- **Error** if none of the above cases can be determined.

For details about the linear regression step, removal of outliers, and how we choose the thresholds, see the extended version of the paper [5].

5 Results

Table 1 shows results from 5 days of data collection, where $S \to C$ is **Server-to-client-dropped**, *None* is **No-packets-dropped**, $C \to S$ is **Client-to-server-dropped**, and *Error* is **Error**. *CN* is China, *Asia-CN* is other Asian countries, *EU* is Europe, and *NA* is North America. For server types, *Tor-dir* is a Tor directory authority, *Tor-bri* is a Tor bridge, and *Web* is a web server.

Our expectation would be to observe **Server-to-client-dropped** for clients in China and Tor servers because of Winter and Lindskog's observation that the SYN/ACKs are statelessly dropped by the "Great Firewall of China" (GFW) based on source IP address and port [8]. We would expect to see **No-packets-dropped** for most web servers from clients in China, unless they host popular websites that happen to be censored in China. Similarly, in the expected case we should observe **No-packets-dropped** for clients outside of China, regardless of server type. We expect a few exceptions, because censorship happens outside of China and because the GFW is not always 100% effective. In particular, Tor bridges are not blocked until the GFW operators learn about them, and some routes might not have filtering in place. Our results are congruent with all of these expectations.

In 5.9% of the client/server pairs we tested, multiple cases were observed in the same day. In some cases it appears that noise caused the wrong case to be detected, but other cases may be attributable to routes changing throughout the day [9]. That the data is largely congruent with our expectations demonstrates

Table 1. Results from the measurement study

Client,Server	$S \to C$ (%)	None (%)	$C \to S$ (%)	Error (%)
CN,Tor-dir	2200 (73.04)	19 (0.63)	504 (16.73)	289 (9.59)
Asia-CN,Tor-dir	0 (0.00)	1171 (96.38)	1 (0.08)	43 (3.54)
NA,Tor-dir	1 (0.07)	1217 (90.69)	49 (3.65)	75 (5.59)
EU,Tor-dir	2 (0.28)	695 (97.89)	2 (0.28)	11 (1.55)
CN,Tor-bri	1012 (58.91)	565 (32.89)	31 (1.80)	110 (6.40)
Asia-CN,Tor-bri	0 (0.00)	626 (80.88)	9 (1.16)	139 (17.96)
NA,Tor-bri	0 (0.00)	657 (78.21)	30 (3.57)	153 (18.21)
EU,Tor-bri	0 (0.00)	313 (78.25)	9 (2.25)	78 (19.50)
CN,Web	28 (2.15)	995 (76.30)	36 (2.76)	245 (18.79)
Asia-CN,Web	1 (0.17)	569 (97.43)	1 (0.17)	13 (2.23)
NA,Web	0 (0.00)	606 (93.37)	0 (0.00)	43 (6.63)
EU,Web	0 (0.00)	305 (90.24)	0 (0.00)	33 (9.76)
All Web	29 (1.01)	2475 (86.09)	37 (1.29)	334 (11.62)
All Tor-bri	1012 (27.12)	2161 (57.90)	79 (2.12)	480 (12.86)
All Tor-dir	2203 (35.09)	3102 (49.40)	556 (8.85)	418 (6.66)

the efficacy of the approach, and some of the data points that lie outside our expectations have patterns that suggest that a real effect is being measured, rather than an error. For example, of the 28 data points where web servers were blocked from the server to the client in China, 20 of those data points are the same client/server pair.

38% of the data we collected does not appear in Table 1 because it did not pass liveness tests. Every 5-minute data point has three associated liveness tests. If a server sends fewer than 2.5 SYN/ACKs in response to SYNs from the measurement machine, a client responds to less than $\frac{3}{5}$ of our SYN/ACKs, or a measurement machine sending thread becomes unresponsive, that 5-minute data point is discarded.

Two out of the ten Tor directory authorities never retransmitted enough SYN/ACKs to be included in our data. Of the remaining eight, two more account for 98.8% of the data points showing blocking from client to server. These same two directory authorities also account for 72.7% of the **Error** cases for directory authorities tested from clients in China, and the case of packets being dropped from server to client (the expected case for China and the case of the majority of our results) was never observed for these two directory authorities.

When Winter and Lindskog [8] measured Tor reachability from a virtual private server in China, there were eight directory authorities at that time. One of the eight was completely accessible, and the other seven were completely blocked in the IP layer by destination IP (*i.e.*, **Client-to-server**). In our results, six out of ten are at least blocked **Server-to-client** and two out of ten are only blocked **Client-to-server** (two had all results discarded). Winter and Lindskog also observed that Tor relays were accessible 1.6% of the time, and we observed that directory authorities were accessible 0.63% of the time. Our results have

geographic diversity and their results can serve as a ground truth because they tested from within China. In both studies the same special treatment of directory authorities compared to relays or bridges was observed, as well as a small percentage of cases where filtering that should have occurred did not.

To evaluate the assumption that clients with a global IPID are easy to find in a range of IP addresses that we desire to measure from, take China as an example. On average, 10% of the IP addresses in China responded to our probes so that we could observe their IPID, and of those 13% were global. So, roughly 1% of the IP address space of China can be used as clients for measurements with our method, enabling experiments with excellent geographic and topological diversity.

6 Related Work

Related work directly related to idle scans [2,3,4] was discussed in Section 1. Other advanced methods for inferring remote information about networks have been proposed. Qian *et al.* [10] demonstrate that firewall behavior with respect to sequence numbers can be used to infer sequence numbers and perform off-path TCP/IP connection hijacking. Chen *et al.* [11] use the IPID field to perform advanced inferences about the amount of internal traffic generated by a server, the number of servers in a load-balanced setting, and one-way delays. Morbitzer [12] explores idle scans in IPv6.

iPlane [13] sends packets from PlanetLab nodes to carefully chosen hosts, and then compounds loss on specific routes to estimate the packet loss between arbitrary endpoints without access to those endpoints. This does not detect IP-address-specific packet drops. Our technique, in contrast, can be used to detect intentional drops of packets based on IP address and requires no commonalities between the measurement machine's routes to the server or client and the routes between the server and client. Queen [14] utilizes recursive DNS queries to measure the packet loss between a pair of DNS servers, and extrapolates from this to estimate the packet loss rate between arbitrary hosts.

7 Conclusion

We have presented a method for detecting intentional packet drops (*e.g.*, due to censorship) between two almost arbitrary hosts on the Internet, assuming the client has a globally incrementing IPID and the server has an open port. Our method can determine which direction packets are being dropped in, and is resistant to noise due to our use of an ARMA model for intervention analysis. Our measurement results are congruent with current understandings about global Internet censorship, demonstrating the efficacy of the method.

Acknowledgments. We would like to thank the anonymous PAM 2014 reviewers and our shepherd, Jelena Mirkovic, as well as Terran Lane, Patrick Bridges,

Michalis Faloutsos, Stefan Savage, and Vern Paxson for helpful feedback on this work. This material is based upon work supported by the National Science Foundation under Grant Nos. #0844880, #1017602, #0905177, and #1314297.

References

1. arma: Research problem: Five ways to test bridge reachability. Tor Blog (December 1, 2011), https://blog.torproject.org/blog/research-problem-five-ways-test-bridge-reachability
2. Antirez: new tcp scan method. Posted to the bugtraq mailing list (December 18, 1998)
3. Lyon, G.: Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning. Insecure.Org LLC, Sunnyvale, CA, USA (2009)
4. Ensafi, R., Park, J.C., Kapur, D., Crandall, J.R.: Idle port scanning and non-interference analysis of network protocol stacks using model checking. In: Proceedings of the 19th USENIX Security Symposium, USENIX Security 2010. USENIX Association (2010)
5. Ensafi, R., Knockel, J., Alexander, G., Crandall, J.R.: Detecting intentional packet drops on the Internet via TCP/IP side channels: Extended version CoRR abs/1312.5739 (2013), http://arxiv.org/abs/1312.5739
6. Alexa: Alexa top 1,000,000 sites, http://www.alexa.com/topsites
7. MaxMind: How accurate are your GeoIP databases? http://www.maxmind.com/en/faq#accurate
8. Winter, P., Lindskog, S.: How the Great Firewall of China is Blocking Tor. In: Free and Open Communications on the Internet. USENIX Association (2012)
9. Paxson, V.: End-to-end internet packet dynamics. SIGCOMM Comput. Commun. Rev. 27(4), 139–152 (1997)
10. Qian, Z., Mao, Z.M.: Off-path TCP sequence number inference attack - how firewall middleboxes reduce security. In: Proceedings of the 2012 IEEE Symposium on Security and Privacy, SP 2012, pp. 347–361. IEEE Computer Society, Washington, DC (2012)
11. Chen, W., Huang, Y., Ribeiro, B.F., Suh, K., Zhang, H., de Souza e Silva, E., Kurose, J., Towsley, D.: Exploiting the IPID field to infer network path and end-system characteristics. In: Dovrolis, C. (ed.) PAM 2005. LNCS, vol. 3431, pp. 108–120. Springer, Heidelberg (2005)
12. Morbitzer, M.: TCP Idle Scans in IPv6. Master's thesis, Radboud University Nijmegen, The Netherlands (2013)
13. Madhyastha, H.V., Isdal, T., Piatek, M., Dixon, C., Anderson, T., Krishnamurthy, A., Venkataramani, A.: iPlane: an information plane for distributed services. In: Proceedings of the 7th Symposium on Operating Systems Design and Implementation, OSDI 2006, pp. 367–380. USENIX Association, Berkeley (2006)
14. Wang, Y.A., Huang, C., Li, J., Ross, K.W.: Queen: Estimating packet loss rate between arbitrary internet hosts. In: Moon, S.B., Teixeira, R., Uhlig, S. (eds.) PAM 2009. LNCS, vol. 5448, pp. 57–66. Springer, Heidelberg (2009)