The Tale of the Weather Worm

Joe Szabo, John Aycock, Randal Acton, and Jörg Denzinger

Department of Computer Science, University of Calgary
2500 University Drive N.W., Calgary, Alberta, Canada T2N 1N4
{szaboj,aycock,acton,denzinge}@cpsc.ucalgary.ca

TR 2007-858-10, April 2007

Abstract. How humans behave when faced with a disaster, natural or man-made, can be exploited automatically by news-aware malicious software. We introduce the idea of *weather worms*, worms that can automatically identify abnormal events and their location, and target computers at that physical location. Such worms could be used to take advantage of poorly-defended computers in a disaster zone, and could even amplify the effects of a physical terrorist attack. Defenses against weather worms require serious examination of policy and presentation of information on the Internet.

Key words: Network security, worms, emergencies, disasters, geolocation, information policy

1 Introduction

The role that humans play in computer security is well-known. Humans are frequently touted as the weakest link in security, and there are always risks of insider attacks, social engineering, and misconfigured and unpatched machines leading to vulnerabilities. These risks result from the behavior of individual humans.

What is not well-known is the effect of large-scale human behavior on security, behavior that large groups of people naturally exhibit at the same time. In previous work, we examined how worldwide combinations of normal business hours and public holidays could yield large windows of vulnerability [1]. This idea can be extended further, as people are also distracted *en masse* by abnormal events like severe weather.

For example, a blizzard could occur in a location which is ill-prepared to deal with one – a blizzard in New York might have the effect of forcing workers to stay home for several days. By staying home, people who are normally on-site, maintaining and patching corporate computers may be less able to do so, and their ability to respond to an attack may be limited as well. Other events might have similar effects; tornadoes, tsunamis, earthquakes, riots, and even transit strikes could result in similar circumstances, either directly by preventing access to computers or indirectly by providing considerable distraction. More sinister is the threat of terrorism, where a physical attack is accompanied by a virtual attack, automatically targeting computers in the affected area.

Obviously these attacks would not always be feasible. For example, a severe weather event may cut power and communications to potential targets (although some large data centers may have their own power sources). This is also recognized in military operations, where a physical attack destroys the infrastructure required to conduct electronic warfare [2]. However, relying on the loss of power and communications as a defense is hardly prudent, and the possibility of the attacks we describe must be considered seriously.

In this paper, we present *weather worms*, worms which are able to automatically detect disruptive events like severe weather, and locate and attack computers in the region. The novel parts of a weather worm, which are not malicious by themselves, have been tested as proofs of concept.

Section 2 discusses the implementation of weather worms. Section 3 looks at defenses against weather worms, which also requires an examination of policy surrounding the availability and presentation of information. Related work is discussed in Section 4, and Section 5 concludes.

2 Weather Worms

For our purposes, a "traditional" worm can be thought of as having four main components: target identification (often called scanning), target infection, trigger, and payload.[1] A weather worm is a variant of a traditional worm, which may manifest itself in two main ways:

Event-based target identification. A weather worm, following its initial seeding, would not spread further, but wait for a major event to occur. Once a major event happened, the weather worm would try to infect machines geographically-located in the affected area. This type of weather worm would try to take advantage of machines being less-maintained while people are distracted by an abnormal event.

Event-based trigger. A weather worm could identify and infect targets through normal means; infected machines would look for a major event to trigger the payload. The payload itself, possibly a DDoS attack, would be focused on machines in the geographic area where the major event occurred. Weather worms of this type would be trying to add to the confusion of a major event by disrupting Internet traffic in the region.

As a special case, one specific physical location (e.g., a world capital) could be targeted by looking for a major event only in that location.

It is important to note the distinction between weather worms and malware like Small.DAM, a.k.a. Storm Worm [3]. In Small.DAM, manually-constructed email subject lines, one even related to a severe storm in Europe [4], are used to entice users to run a malicious email attachment. A weather worm, in contrast, *automatically* identifies important news events and their location, and *automatically* finds targets at the location to attack.

[1] The trigger and payload are optional.

There are two novel technical aspects to a weather worm. First, Internet information sources are used to identify events and locations, resulting in what we call *news-aware* malicious software. Second, IP addresses of potential targets in a given location are found through a process we call *inverse geolocation*. In both cases, we have looked for lightweight solutions that might be used by a worm author, which gives us some insight into the possible scope of such a threat and effective defenses. The remainder of this section discusses these lightweight solutions in detail.

2.1 Identifying Events and Locations

The Internet offers a plethora of data sources that might be useful for identifying events and locations, though not all sources are suitable for this particular task. If we take weather as an example, depending on the location, the same two weather events may have a drastically different impact. A blizzard can be a regular occurrence in northern Canada, and the event would have far less impact than the same blizzard occurring in New York. While weather sources would report both events, news sources would only choose to report the more significant of the two, making news services a better source for this particular application. In addition, using multiple news sources allows measurement of the importance of a particular event. A large number of news sources reporting on the same event might indicate that the event in question is actually very important and could possibly impact a large number of people. News sources also permit the identification of events to extend beyond weather. We see malicious software that makes use of news sources, i.e., news-aware malicious software, as a potential threat to computer security.

In general, our problem of event identification and location has to be considered an application within the area of information retrieval on the Internet (see [5]). Information retrieval is a very active research area and clearly Internet search engines are an example of successful applications of methods from this research area. For the purpose of this paper, we can think of information retrieval as an area that tries to achieve some limited understanding of (English) texts without doing any real natural language processing (in the sense of, for example, [6]). In the following, we describe a system that uses only straightforward instantiations of information retrieval techniques that are easy to implement. While these instantiations come with known problems, that we will discuss, there are a number of more sophisticated techniques that do not suffer from these problems, but might require more heavyweight implementations.

Our prototype system, NewsWatcher, requires input from three separate files: a "seed" URL list, an event list, and a location list containing city names. The seed URL list specifies starting URLs where NewsWatcher can begin its news search. The event list provides event names for the system to search for, and match to locations provided in the location list. Appendix A gives the contents of these lists in more detail.

URL Processing. NewsWatcher operates by processing a queue of URLs. Initially, the queue consists of the seed URLs. Each URL in the queue is processed in three steps:

1. Downloading. The URL content is downloaded, but only if the content size is under 150 K; this prevents NewsWatcher from downloading large media files instead of web pages. The `<head>` portion of downloaded HTML content is discarded, because metadata there was found to list many event types that would otherwise cause false matches.
2. Link detection. Each HTML page is scanned for links to add to the URL queue. Some links are ignored outright: self-referencing URLs, URLs that use JavaScript, `mailto:` URLs, and URLs that contain the string "blog." Because many news sites tend to archive old pages, URLs that contain the string "archive" are ignored. URLs are also ignored that have already been seen by NewsWatcher.

   Subject to these constraints, NewsWatcher traverses all links found on the seed URL pages. Links on subsequent pages are only followed if the (human-readable) URL link text contains an event name. This allows traversing a second level of links, but only when the link appears to be of interest.

   At no time does NewsWatcher follow more than two levels of links from a seed URL, and may only traverse one level deep. This happens for two reasons. First, important news stories will not be buried in a news website. Second, our experiments showed that regularly traversing beyond one level of links yielded an unmanageably large number of new URLs; with our current implementation and input files, we saw a comfortable average of 3709 URLs processed.
3. Event searching. The HTML page is searched for all events provided in the event list. If an event is found in the page, NewsWatcher defines a *search field* around that event name in which it searches for any matching location names.

   The search field size is guided by statistical properties of English. Measures of the average English sentence length range from 16.5 words [7] to 19.3 words [8]; we used the conservative value of 20 words. Almost 70% of English words have five characters or less [8]. Combining these two measures implies that the average English sentence is no more than 100 characters long – we thus used a search field containing 100 characters on both sides of the event name. If periods are found within the search field on either side of the event name, then the search field is truncated at that point, because we only want to examine the sentence in which the event name is found.

   If a location is found in the search field, the event and its location are stored for later output.

Once these steps are complete, the URL is fully processed, and NewsWatcher continues onto the next queued URL until the queue is empty.

Problems. Several complications were identified during the design and development of the current version of NewsWatcher.

Problem 1. How is the system to discern whether a news story is reporting an event that is currently happening, or that had occurred a year earlier (i.e., a retrospective anniversary)?

This issue was mostly addressed by ignoring event matches whose corresponding search field were found to contain the words "ago" or "anniversary." Further problems of this nature were also averted by ignoring URLs that contained the string "archive." This problem is difficult to overcome entirely, as some news stories reference prior events while reporting current events.

Problem 2. In cases such as hurricanes, where news services are able to report the event long prior to it actually occurring, will it be possible to distinguish between event buildup and the event itself?

Over the course of an event, it was found that the output from the prototype actually can be used to identify event buildup and cool-down periods. Events that originally started with one or two URL matches would have many more URL matches in later runs. Later still, there would again be few URL matches for the event.

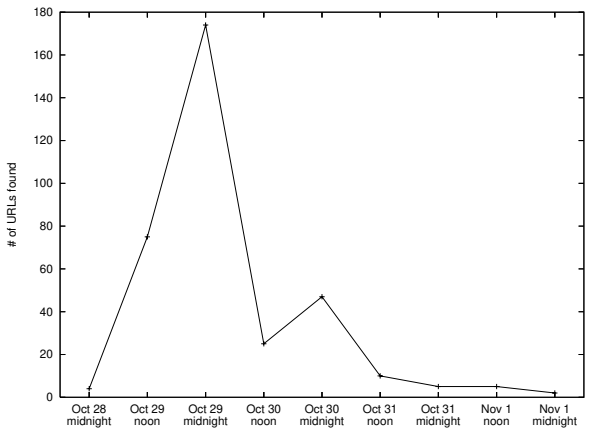

Fig. 1. Number of URLs over time reporting on violence/riots in Marseille, 2006

For example, Fig. 1 shows the number of violence and riot events detected in Marseille over a number of days; the buildup and cool-down periods for the event are apparent. It is possible to define URL match thresholds (discussed in the next section) to estimate the current status of an event based on the number of URLs found. For example, in the above graph, a number less than 100 URL matches

could be considered to be either buildup or cool-down periods. This approach could be rendered ineffective if more important events overshadow others, even though they might still be in the height of their buildup.

Without more closely examining web pages for event occurrence times and dates or close examination of sentence structure or word usage, it may not be possible to identify such periods in single runs of the algorithm without jeopardizing the lightweight nature of NewsWatcher.

Problem 3. Name confusion has to be accounted for. The system must be able to discern whether names found in news stories such as the Carolina Hurricanes or Calgary Flames[2] are actually events.

An easy solution to this problem is to require an exact match with the names in the event list and to avoid the use of event names such as "Hurricanes" or "Flames."

Problem 4. The system must be as small and lightweight as possible, to compare meaningfully to one that might be bundled in the (small) package of a worm.

The set of input files for NewsWatcher contains 6416 location names, 38 event names, and 24 starting URLs. The total size of these text files is under 64KB and they can be easily compressed to as small as 29KB. The size can further be reduced by having fewer numbers of locations in the location list; in either case the size of the data is insignificant, especially considering the majority of users who enjoy high-speed Internet connections today.

Problem 5. Events can span many degrees of severity. Is it possible for the system to be able to determine if an event is still in its infancy, or if an event is too great that all computers in the corresponding location would be either destroyed or unable to infect because of the extreme circumstances? In both cases, having the event reported would be of little use.

This problem is very similar to that of identifying event buildup and cool-down periods and the aforementioned observations also apply here. Similarly, without implementing more complex algorithms which could consider sentence structure and word usage, this problem is also quite similar to solve. A full solution could (again) jeopardize the lightweight requirement.

To partially address the issue, however, some modifications to NewsWatcher could be made. It is possible to implement an additional input list containing "blacklist" events. Finding a blacklisted event would cause any other event matches in the search field to be ignored.

Suppose "volcano erupted" was contained in the event list and "buildings destroyed by lava" was contained in the event blacklist. If a news article read "the volcano erupted on Tuesday, and the resulting devastation left hundreds of buildings destroyed by lava," the "volcano" event would be ignored. Similarly, if "hurricane" was in the event list and "expected to arrive" were in the event

[2] The Carolina Hurricanes and the Calgary Flames are both professional hockey teams.

blacklist, no events would be matched in the text "hurricane Bob is expected to arrive next week."

This method would ultimately eliminate all such matches, or at least reduce them so that the output reports fewer URL matches for an event, thus reducing the apparent importance of the event from NewsWatcher's point of view.

Problem 6. Some event names are also names of certain objects, and in some contexts it can be impossible to distinguish between actual events and other non-related reports.

Problem 7. Some event names can be used to metaphorically describe totally unrelated events.

The last two problems have been encountered during testing. Consider these excerpts from actual news web pages:

> 'War games: Tornado jets zoom out of Calgary as British troops train for combat' [9]

from which NewsWatcher identifies the event "Tornado" at location "Calgary," and

> 'Tuesday's electoral earthquake triggered an equally seismic reaction in Washington yesterday, ...' [10]

mistakenly identifies an "earthquake" in "Washington."

Though a possible solution might involve the aforementioned event blacklist (in this case blacklisting the word "jets" would have prevented the first problem), for solving these problems it might be necessary to use more complicated sentence or pattern recognition techniques from information retrieval (see [5]). As for metaphors, even relatively recent work in computational linguistics 'is far from being able to recognize metaphoric language in general' [11, page 43].

Interpreting NewsWatcher Output. NewsWatcher's output is a list of triples, abstractly speaking, where each triple identifies an event, a location, and the number of URLs the match was found at. How should this output be interpreted?

If a greater number of event matches are desired, at the expense of some incorrect matches, the most straightforward interpretation of the output is to simply regard the events reported as correct and target all of the locations listed in the output. Unfortunately, NewsWatcher is not accurate enough to allow its output to be safely regarded in this way. For example, "violence in Washington" showed up frequently but was incorrectly detected; news articles were found that report on people in Washington discussing violence elsewhere in the world.

While some incorrect matches can be tolerated, it is generally the case that very few URL matches for an event implies the event itself is either not very important, or is not an actual event and may simply be due to one or more problems discussed in the previous section.

To address this problem, a threshold value can be selected: only the events having a number of URLs higher than the threshold value would be considered correct. Choosing a single, static threshold value is difficult, though, because there is a normal variation in news volumes. Intuitively, this can be thought of as how "busy" the news day is.

Table 1. Top three events by URLs matched on slow/moderate/busy news days

	Event	Location	# URLs
Slow news day	tornado	Carolina	17
	violence	Baghdad	16
	floods	Kabul	11
Moderate news day	flooding	Nairobi	38
	violence	Washington	21
	violence	Baghdad	19
Busy news day	fires	Oaxaca	60
	violence	Baghdad	36
	violence	Washington	25

Table 1 illustrates the difference in terms of the number of URLs reported. Although the full sets of triples are omitted for space reasons, a static threshold of 12 would eliminate much of the noise regardless of news volume. Unfortunately, depending on how busy the news day is, a static threshold might result in very few event matches and might actually filter out wanted data. Using the "slow news day" data set as an example, a threshold of 12 eliminates the "floods for location Kabul" event (because only 11 URLs were found reporting the event), whereas this event is detected correctly and is potentially of interest.

An alternative to using a static threshold is implementing a dynamic threshold with respect to the output, such as only using the top k triples with the largest number of URLs. Again, this has problems, because even the most-reported events on a slow news day are probably not cataclysmic enough to result in the human distraction the weather worm tries to take advantage of.

We conjecture, based on our experience with NewsWatcher, that the likeliest implementation would use a combination of static and dynamic thresholds. A static threshold would help filter spurious matches and weed out correctly-detected but minor events; this would probably be no lower than 50 URLs. A dynamic threshold would pick out the top remaining major events, if any. Invariably some tuning of the static threshold would be necessary.

NewsWatcher clearly demonstrates that it is possible to detect news events and their locations in a lightweight fashion. While its detection is not perfect, malware does not need to operate with complete precision, and we conclude that this aspect of a weather worm is already technically feasible.

2.2 Inverse Geolocation

By identifying events and locations, a worm author could build an automated version of the "Storm Worm" [3]. No need to manually distribute worm variants with new, enticing headlines: the worm itself would identify important headlines and modify its infectious emails accordingly. Indeed, this is a possible next step in worm evolution.

The situation is worse, however, if a full weather worm is developed. This would allow worm activity to be targeted to an event-affected region using information that is already available to worm authors. We have published a full account of how to identify computers in a geographical region elsewhere [12], but we briefly summarize the results here for completeness.

Geolocation is the process of mapping an IP address into a physical location; a full survey of geolocation techniques may be found in [13]. We refer to the inverse process as *inverse geolocation*: mapping a physical location into IP addresses. In terms of a weather worm, when an event is detected and located, what are the IP addresses of potential targets?

There are two main ways we have demonstrated that inverse geolocation is possible:

1. A database intended for geolocation was transformed for use with inverse geolocation. By introducing some small errors (which would be acceptable for malware), and limiting the locations to cities with over 1,000,000 people, our inverse geolocation database was compressed to just over 700 K.
2. Using well-known Internet services – a search engine and the *whois* database – we were able to produce large lists of IP addresses for specified locations.

Using either approach, a weather worm could build a list of potential targets in a relatively lightweight fashion. We conclude that this aspect of a weather worm is also technically feasible.

3 Defending Against Weather Worms

There are four primary avenues to consider for defending against weather worms and other news-aware malicious software.

First, there is the usual suspect. Assuming a weather worm sample had been acquired, anti-virus software is probably the only defense common enough to eradicate an established weather worm prior to it triggering. Curiously, it is more important in this case for all machines *except* targeted machines to be running updated anti-virus software. The machines that will be targeted will not be known in advance, however, meaning that all machines should be running anti-virus software as usual.

In the event of a highly-targeted attack by news-aware malicious software, such as coordinating virtual and physical terrorist attacks, capturing a sample before the triggering event may be critical. A sample may reveal the site of a planned physical terrorist attack by virtue of an unusually-short location list, allowing proper precautions to be taken.

Second, reliance on human maintenance of computers must be reduced. For example, automatic patching of machines might be considered [14]. Normally this is suggested because of concern over worms which spread extremely quickly, like "flash worms" [15, 16]; humans are unable to react to such worms in time. The weather worm concept suggests that less reliance on humans is good not because humans can't react in time, but because they can be distracted by a major event and rank other priorities (e.g., family, personal survival) more highly then computer security.

Third, the weather worm can be prevented from gathering the information it needs to operate. The weather worm primarily draws upon news web sites; HTTP access is extremely unlikely to be blocked, and there are no telltale search engine queries to detect and block either, so any defense has to be from the information source itself.

Information sources can be blocked from fully-automated use entirely. This is currently accomplished by applying Turing tests [17], perhaps better known as CAPTCHAs [18], tests which are easy for a human to solve[3] but hard for a computer to solve.[4] Widespread use of Turing tests for accessing simple news websites would be incredibly annoying, however, and would break legitimate automatic news programs, so we do not see Turing tests being a very effective defense.

The results from information sources can be made harder to interpret automatically. As Section 2.1 showed, the current version of the NewsWatcher prototype has several limitations that render it ineffective against web pages having any of the following characteristics:

- The page identifies event and location names in separate sentences, or at least 200 characters apart.
- The page and its content is greater than 150,000 bytes in size (larger files are ignored by NewsWatcher).
- Event or location names are spelled with s p a c e s in them, or spelled incorrectly.

These are defenses that exploit specific, known problems in NewsWatcher. To defend against news-aware malicious software in general, as a news source, the following techniques can be useful:

- Using frames to display the event and location names from distinct HTML source pages.
- Displaying data in a non-text form, representing events and locations using media such as image files or Flash.
- Using any form of HTML obfuscation [19].
- Changing the words and phrases used to describe events, and using more ambiguous words and metaphor.

[3] In theory.
[4] Also in theory.

But all these techniques come with costs and there is already known research in information retrieval that can make the techniques useless. The first three general techniques result in news pages that are larger, thus putting more strain on networks. In addition, they make it more difficult for the human user to use tools that help with the classification of news for non-malicious purposes. This is also true for the fourth general technique, which additionally might make it difficult for the user to understand the intended information easily. In fact, a common expectation of human users is to get news in a concise, clear, and easy-to-understand form from news web sites.

There are already tools available that convert even complex HTML pages to standard text pages and converting pictures containing easy-to-read text to ASCII text, while not a lightweight task, nevertheless is an area where there are already tools available. If some computing effort is invested, even spelling mistakes and spacing of characters are no longer big problems.

Technical and usability considerations aside, there is currently no compelling reason for news sources to alter their output to be hostile to weather worms. We therefore would not rely on this as a defensive measure in practice.

The fourth and final avenue of defense is on the side of the targeted computers. Countermeasures must include the possibility of a news-aware attack in the emergency plans that people, businesses, and other organizations have. If it is not possible to shut down the machines during the period of an emergency then the following measures could be taken:

- Transfer necessary offered services to better supervised computers in unaffected areas.
- Arrange for intensified remote supervision of the computers in the emergency area.
- Use stricter security policies for the duration of the emergency that might deny some legitimate requests but block the attack or its effects. Under normal circumstances, learning techniques could be used to discern the standard behaviors of users, and then allow only requests following the learned pattern precisely. In an emergency situation, however, the legitimate traffic pattern will almost certainly be abnormal, and so learning techniques would be less effective.

Overall, a thorough disaster recovery plan should require little change to adapt to news-aware malicious software attacks. The key realization is perhaps simply that such attacks are possible, and to be prepared for them.

4 Related Work

The related work can be divided into three parts. We discuss work related to NewsWatcher and the overall idea of weather worms in this section; work related to inverse geolocation will not be duplicated here, but may be found in [12].

4.1 NewsWatcher-Related Work

As already mentioned, Internet search engines like Google (see [20]) employ techniques related to our current version of NewsWatcher. NewsWatcher can be viewed as performing very simplified web crawling. There are many other systems whose goal is to overcome the information overload that the Internet offers to users. [21] provides users with summaries of multiple news articles. The build-up detection for an event by NewsWatcher can be seen as a crude summary.

NewsWatcher achieves the coordination of a group of malicious programs in a natural manner, by having them process the same news information. The potential of using the Internet for coordinating software was pointed out in [22]. But NewsWatcher adds to this some flexibility by not requiring precisely defined event descriptions but essentially using event types and achieving a kind of opportunism in its target selection. The fact that communication between programs can be avoided if the programs share the exactly same information about their (perceived) environment and use the same decision making algorithms was first pointed out in [23].

Several techniques have been previously suggested for identifying real-world locations referred to by documents, including web pages. Special attention is paid to resolve place name "disambiguation" [24]: resolving the difference between ambiguous place names such as the city named Batman in Turkey, or the comic-book character named Batman. This is also referred to as geo/non-geo ambiguity [25]. To further complicate the problem, one must also address geo/geo ambiguity, or the ambiguity caused by different places sharing the same name. Another closely related issue is 'the problem of indexing and navigation of web resources by geospatial criteria,' [26, page 221] and the problem of location aliasing (for example, L.A. instead of Los Angeles).

NewsWatcher's lightweight approach precludes carrying the amount of data or performing the level of computation required by other proposed solutions. The NewsWatcher prototype location list contains only locations having large populations, which removes most ambiguous location names. Beyond that, we found that the best way to address ambiguity was to manually remove offending location names, making these locations immune to NewsWatcher.

The problem of geo/geo ambiguity is ignored completely; NewsWatcher is more concerned with gathering location *names*, and not the actual whereabouts of the location in question. The location aliasing problem rarely occurs on news web sites; during the prototype testing, the news sites that were encountered always tended to use the full location names (although state names were commonly abbreviated).

4.2 Weather Worm Work

The underlying threat of having such great amounts of information in computer-accessible form can be hard to comprehend. Byers et al. 'demonstrate that the current Internet services offer an avenue of attack against physical world processes' [27, page 240]. They describe a specific attack scenario through/on the

physical postal service, launched from the Internet using publicly-available information. Although they do not cause a direct physical effect, weather worms are an example of an attack using publicly-available information, which can focus on Internet-only targets, or amplify the effect of preexisting physical events.

HTML obfuscation is of course well-known to spammers [28]. Using HTML obfuscation and Turing tests as a *defense* has been suggested previously [22, 27], although we doubt their overall usefulness against weather worms.

The abuse of search engines and the information they provide was pointed out by Byers et al. [27]. Weaver et al.'s worm taxonomy noted the possibility of using search engines to identify vulnerable targets [29], an idea expanded upon by Provos et al.'s "search worms" [30]. We know of only one example of this occurring in the wild: Santy [31]. However, Santy and the previous work deals with searching for targets exploitable using some specific vulnerability.

While worm activation through human activity has been mentioned [29], the activities noted were those of individual humans. The only work examining large-scale human behavior we are aware of is our own [1].

5 Conclusion

The weather worm shows that news-aware malicious software is possible, software that automatically notes a major event, identifies the location, and finds targets at the physical location of the event. Human behavior, the mass distraction of a major event, might thus be exploited maliciously.

While defenses, both specific and general, can be deployed by information providers, it is unlikely that they will be deployed unless news-aware malware becomes a frequent threat. Instead, the onus is on public- and private-sector organizations – potential targets – to incorporate countermeasures into their disaster plans.

Generally, we are forced to conjecture that any information provided for legitimate purposes can be used for malicious purposes. The very information and infrastructure we rely upon as a society is the information and infrastructure through which society can be attacked.

References

1. Friess, N., Vogt, R., Aycock, J.: Timing is everything. Computers & Security **24**(8) (2005) 599–603
2. Headquarters, Department of the Army: Information operations. Field manual No. 100-6 (27 August 1996) United States Army.
3. F-Secure: Small.DAM. F-Secure Trojan Information Pages (17 January 2007)
4. Rising, D.: Storms in Europe kill 46, disrupt travel. Associated Press (19 January 2007)
5. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2007)
6. Jurafsky, D., Martin, J.H.: Speech and Language Processing. Prentice-Hall (2000)

7. Flesch, R.: A new readability yardstick. Journal of Applied Psychology **32**(3) (1948) 221–233
8. Kučera, H., Francis, W.N.: Computational Analysis of Present-Day American English. Brown University Press (1967)
9. Calgary Herald: War games: Tornado jets zoom out of Calgary as British troops train for combat. http://www.canada.com/calgaryherald/news/city/-story.html?id=32dad626-2a7d-497b-a369-64e2f68b91a3 (2006) Page as of 11-11-06.
10. Balz, D.: For Bush's new direction, cooperation is the challenge. http://www.washingtonpost.com/wp-dyn/content/article/2006/11/08/-AR2006110800489.html (2006) Page as of 11-11-06.
11. Mason, Z.J.: CorMet: A computational, corpus-based conventional metaphor extraction system. Computational Linguistics **30**(1) (2004) 23–44
12. Acton, R., Friess, N., Aycock, J.: Inverse geolocation: Worms with a sense of direction. In: Malware '07. (2007) To appear.
13. Muir, J.A., van Oorschot, P.C.: Internet geolocation and evasion. Technical Report TR 06-05, School of Computer Science, Carleton University (2006)
14. Vojnović, M., Ganesh, A.: On the effectiveness of automatic patching. In: Proceedings of the 2005 ACM Workshop on Rapid Malcode. (2005) 41–50
15. Staniford, S., Paxson, V., Weaver, N.: How to 0wn the Internet in your spare time. In: Proceedings of the 11th USENIX Security Symposium. (2002)
16. Staniford, S., Moore, D., Paxson, V., Weaver, N.: The top speed of flash worms. In: Proceedings of the 2004 ACM Workshop on Rapid Malcode. (2004) 33–42
17. Naor, M.: Verification of a human in the loop or identification via the Turing test. http://www.wisdom.weizmann.ac.il/~naor/PAPERS/human_abs.html (1996)
18. von Ahn, L., Blum, M., Hopper, N.J., Langford, J.: CAPTCHA: Using hard AI problems for security. In: Proceedings of EUROCRYPT 2003 (LNCS 2656). (2003) 294–311
19. Graham-Cumming, J.: The spammer's compendium. (http://www.jgc.org/tsc/)
20. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems **30**(1-7) (1998) 107–117
21. McKeown, K., Radev, D.R.: Generating summaries of multiple news articles. In: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. (1995) 74–82
22. Lee, H.H., Chang, E.-C., Chan, M.C.: Pervasive random beacon in the Internet for covert coordination. In: Information Hiding, 7th International Workshop, IH 2005 (LNCS 3727). (2005) 53–61
23. Pynadath, D.V., Tambe, M.: Multiagent teamwork: Analyzing the optimality and complexity of key theories and models. In: Proceedings AAMAS 2002, Bologna (2002) 873–880
24. Zong, W., Wu, D., Sun, A., Lim, E.-P., Goh, D.H.-L.: On assigning place names to geography related web pages. In: JCDL '05: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries. (2005) 354–362
25. Amitay, E., Har'El, N., Sivan, R., Soffer, A.: Web-a-where: geotagging web content. In: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and development in Information Retrieval. (2004) 273–280
26. McCurley, K.S.: Geospatial mapping and navigation of the Web. In: Proceedings of the 10th international conference on World Wide Web. (2001) 221–229
27. Byers, S., Rubin, A.D., Kormann, D.: Defending against an Internet-based attack on the physical world. ACM Transactions on Internet Technology **4**(3) (2004) 239–254

28. Spammer-X: Inside the SPAM Cartel. Syngress (2004)
29. Weaver, N., Paxson, V., Staniford, S., Cunningham, R.: A taxonomy of computer worms. In: WORM '03. (2003) 11–18
30. Provos, N., McClain, J., Wang, K.: Search worms. In: WORM '06. (2006) 1–8
31. Hyponnen, M.: F-Secure virus descriptions: Santy (2004)

Acknowledgments. The research of the second author is supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada. Nathan Friess performed the inverse geolocation database experiments.

A NewsWatcher Inputs

Event List

```
wildfire
wild fire
fires
wildfires
wild fires
tornado
tornados
earthquake
earthquakes
tsunami
hurricane
transit strike
transit strikes
power outage
power outages
brownout
brownouts
riot
riots
blizzard
blizzards
quake
quakes
severe weather
snowstorm
snowstorms
snow storm
violence
without electricity
without power
without water
thousands injured
thousands killed
tropical storm
tropical storms
flooding
floods
typhoon
```

Seed URL List

```
http://news.google.com/?topic=w
http://msnbc.msn.com/id/3032507/
http://www.cnn.com/WORLD/
http://news.bbc.co.uk/
http://www.cbsnews.com/sections/
  world/main202.shtml
http://www.worldnews.com/
http://www.foxnews.com/
http://today.reuters.com/news/home.aspx
http://news.yahoo.com/
http://www.guardian.co.uk/worldlatest/
http://www.worldnetdaily.com/
http://www.wnnetwork.com/
http://abcnews.go.com/WNT/
http://news.independent.co.uk/world/
http://www.stuff.co.nz/world_news.html
http://www.washingtonpost.com/wp-dyn/
  content/world/index.html
http://www.arabworldnews.com/
http://www.worldpress.org/
http://www.news.com.au/
http://www.sky.com/skynews/
http://www.topix.net/world
http://www.unison.ie/worldnews/
http://www.globaldisasters.com/
http://www.cbc.ca/news/
```

Partial Location List

```
   ⋮
Cologne
Pulheim
Dunkirk
Calgary
Dresden
Horsham
Crawley
Rossosh
Ashford
Legnica
Antwerp
   ⋮
```