

# Security Aspects of Homogeneous Environments

Hanuš Adler\*

Actinet Informační systémy, s.r.o.

U Bulhara 3, Praha 1, Czech Republic

July 9, 2003

Rapid development of computer technologies over the last decade, together with the rise of interconnected networks among which Internet is the most prominent, brought along major problems that were not anticipated, or were largely underestimated, at the time when currently popular software packages were being developed. While the problems, or design flaws, have always caused some damages and much grief to the affected organizations, only in the last few years they became a global phenomenon costing millions of dollars in damages worldwide.

The fast growth of the Internet and the fact that never before had so many people such an immense potential to communicate with other people anywhere in the world is almost certainly the primary cause for this development. Nevertheless, it is quite clear that it is significantly aided by the fact that the computing environment people are commonly using is now nearly homogeneous all over the planet, and that the quality of this environment is undoubtedly well below reasonable expectations.

Over the recent years, computer industry has gone a long way towards uniformity and homogeneity. While it may have seemed desirable for many mainly economic reasons, it has proved rather dangerous from many other viewpoints. Today, even the economic reasons are questionable.

This contribution aims to describe the problems and dangers associated with homogeneous environments, and to argue that the currently popular worldwide usage of such environments causes significant damages and should be avoided whenever possible. Possible countermeasures and solutions now available will be also discussed.

---

\*This paper was originally presented at the Security and Protection of Information conference at IDET 2003

## 1 Introduction

At approximately 2:00 PM GMT-5 on Friday March 26 1999 we began receiving reports of a Microsoft Word 97 and Word 2000 macro virus which is propagating via email attachments.

—CERT Advisory CA-1999-04

On the morning of Friday, March 26, 1999, hardly any common computer user was aware that there was something called a macro virus. Three days later, everyone felt like an expert on them, because of the constant attention TV and other media were giving to the Melissa macro virus over the weekend.

While Melissa was not the first such virus, and wasn't even particularly mean, there was something that made it different from previous ones, and one of the most successful viruses to date. The novelty was simple and elegant—the virus just took some 50 addresses from user's address book and mailed itself to those addresses with a subject that was likely to attract the recipient to click on it.

How was that possible?

The author did not have to search for addresses in all the various places mail clients could be storing them on the hard disk, nor dissect various text or binary formats that they could be using. It was enough to rely on the prevalence of a single operating environment and one application suite with its internal programming language. Using the macro language, it was easy to develop a virus that would retrieve victim's e-mail contacts and re-send itself to them.

Melissa's author simply recognized the right moment at which a great percentage of corporate users were forced to use the same software for documents and communication, and general awareness of security issues was low. Microsoft Windows, Microsoft Office, Outlook, Outlook Express, Exchange were becoming a de facto standard on most desktops throughout corporations, and were easily available to home users as well, which enabled virus authors to develop and test their code without needing access to special HW or SW that would normally be unavailable to home users.

What also contributed to the success of Melissa was that although in 1999 there was still a significant number of people that used a different mail client, thanks to Microsoft's eager pushing of their products in all encompassing bundles, even those who didn't actually use Outlook often had it installed as part of Microsoft Office on their machines—ripe for misuse.

Microsoft Visual Basic that became part of the Office suite provided tools that enabled practically everyone to carry out most complex tasks with their computers without actually having a good knowledge of programming in assembler or C.

Finally, the Internet provided an environment where a virus or worm could propagate rapidly using standardized protocols for information exchange—SMTP being the first, peer-to-peer file sharing and chat protocols like SMB, ICQ and others to follow.

Melissa was just the first virus that successfully demonstrated the dangers of homogeneity, connectivity and programmability<sup>1</sup> put together in the commonly available combination of the ubiquitous Wintel platform, Microsoft application suite and the Internet.

## 2 Dangers

Homogeneity, connectivity, programmability.

—Carey Nachenberg, Symantec Corp., 1999

### 2.1 Workstations

#### 2.1.1 Wintel

“There are no significant bugs in our released software that any significant number of users want fixed.”

—William H. Gates III, Microsoft Corp., 1995<sup>2</sup>

To define the dangers threatening today’s popular computing environments, we must identify their vulnerable parts.

Various estimates suggest that some 90% of common user workstations run one of the Microsoft Windows family of operating systems on Intel hardware platform. These are mostly fully equipped PC’s, not thin clients. On practically every such workstation, there is a copy of Microsoft Office used for creating and reading documents, spreadsheets, presentations etc. Microsoft Outlook (or Outlook Express) is commonly used for e-mail communication, and most of their users also use Microsoft Exchange server for groupware tasks.

Let’s call this environment “Wintel” in the following text.

Most of the Wintel workstations, especially in corporate environments, are interconnected with each other, with internal servers, partner networks, and with the Internet. Also homes now often have some Internet connection, and the number of permanently connected home users steadily grows as affordable permanent connection becomes available through Cable TV, ADSL, Freenets and other community networks.

There are several important problems associated with Wintel environment:

1. it is marketed as easy to use, best choice for less knowledgeable users, and those who do not want to know anything about the tools they are using,

---

<sup>1</sup>See “The Evolving Virus Threat” by Carey Nachenberg at <http://csrc.nist.gov/nissc/2000/proceedings/papers/019.pdf>

<sup>2</sup>See <http://www.cantrip.org/nobugs.html>

2. it hides important information from the user,
3. it commonly does important things without asking the users, behind their backs,
4. on the other hand, it is perpetually asking the user to confirm reading most unimportant info messages and actions,
5. depending on the OS version, it may have none or little security mechanisms implemented,
6. bug fixes are not readily available (some bugs are never fixed, other fixes may be released with considerable delay)
7. security mechanisms implemented on the recent OS versions are rarely used correctly (or at all),
8. it is mostly backward compatible even for MS DOS based applications,
9. practically every machine can be expected to run a similar set of mostly Microsoft applications whose lack of security was repeatedly demonstrated in the past,
10. over-integration of applications eases misuse,
11. its applications often make it extremely easy for a user to run untrusted code,
12. users are encouraged to use dangerous formats and ways for information interchange.

As a result, people are showing a tendency to

1. refuse to learn how to use their computers and the Internet in a secure manner, having been lead into belief that MS Windows and MS applications are so easy no education is necessary,
2. ignore important messages (Windows require the user to click “Yes” or “OK” buttons so often that after a while no user is actually reading the text of the messages any more),
3. ignore suspicious behaviour of their system,
4. ignore security-related information sources and importance of bug fixes,
5. freely run executables and open documents received by e-mail without any real idea whether they can securely identify the sender and the sender’s trustworthiness
6. carry out normal work as users with administrator privileges
7. send zips, crypted files as executable files, and write e-mails in Word or text/html instead of text/plain, which consequently lowers general recipient awareness of the dangers associated with such formats in e-mails.

Although most of the corporate networks employ security administrators, have some kind of a security policy in place and make great effort to implement security measures protecting their users from attacks, all the security effort can be easily thwarted by a simple action of an unsuspecting, naïve Windows user.

Although recent Windows versions can implement packet filters and personal firewalls for Windows are on the rise, home computers are still often unprotected, and even behind the personal firewalls, vulnerable to many attacks.

## 2.1.2 Other Platforms

“Whip me. Beat me. Make me maintain AIX.”

—Stephan Zielinski

Number of non-Wintel platforms on users' desktops is rather low. Largest share probably goes to Apple Mac, however thanks to the nature of Linux and similar systems, which can be downloaded from the Internet for free and used on an unlimited number of computers, it is quite impossible to be quite sure of that.

Also, though most of these systems, including MacOS, are now Unix-based, they substantially differentiate between each other on many accounts. Even those systems that stem from the same foundation, like Linux systems, are often quite different from distribution to distribution, or even from one to another installation.

These systems were designed as multi-user from the very start, and even though Unix security isn't perfect, user space programs would normally only affect files and programs owned by the UID<sup>1</sup> with whose rights they are executed. As it is very uncommon for a Unix user to own any programs, only the affected user's data could be compromised or destroyed.

There are few applications that would be used by every Unix-based system user, and even though some are very popular, like desktops (Aqua, KDE, Gnome and CDE) and browsers (Mozilla and other Gecko based ones, KDE's Konqueror, Apple's Konqueror-based Safari), none of them is actually used by an overwhelming majority of users, although some plugins like Flash are a possible threat for all of their users. As for e-mail clients, Mozilla, Evolution, KMail, Mutt and Pine seem to be among the leading programs, but none is used by a substantial majority of users.

While there are quite a lot of Office suites for Unix based systems, most people are using either Microsoft Office (which is only available for MacOS) or OpenOffice.org, which is largely compatible with MS Office. Other popular packages include T<sub>E</sub>X (L<sup>A</sup>T<sub>E</sub>X), Adobe Acrobat Reader or xpdf. Although MS Office and OpenOffice.org are probably the most vulnerable entry point to the non-Wintel world, they are not used by everyone either.

---

<sup>1</sup>UID = User ID, an account number on Unix-based systems

Currently, no universally applicable threats to the non-Wintel platform workstations are known—partly because of the much smaller number of desktop users, but also because these platforms present a much wider spectrum of applications in common use than the Wintel platform, there are many different processors in use, and last but not least also because users of these platforms (with the exception of MacOS users) nowadays are more knowledgeable than Windows users, which makes them less vulnerable to some forms of attacks.

## 2.2 Servers

### 2.2.1 Wintel

“We don’t do a new version to fix bugs. We don’t.  
Not enough people would buy it.”

—William H. Gates III, Microsoft Corp., 1995<sup>1</sup>

The situation is slightly different on the server market. First of all, servers are maintained by administrators who usually have at least some knowledge of security. Servers accessed from untrusted networks normally do not run applications like MS Outlook or MS Office,<sup>2</sup> and do not usually run too many server applications at one server.

Nevertheless, the applications that are normally run are easy to guess on a great majority of Wintel servers:

- MS IIS for WWW and FTP servers
- MS IIS for a webmail frontend to MS Exchange
- MS Exchange for e-mail transfer and groupware functionality
- MS SQL or MSDE as database backends
- MS DNS server (based on bind)

These applications are often too complex for the intended task, and over-integrated with the system or with each other in many quite unexpected ways.

For example, Microsoft’s IIS is often used for Web servers, but includes also FTP server and Gopher server in one large program. One of the attacks on Web served from IIS servers actually made use of a DoS against the FTP server code. If the servers were programmed as separate programs, the worst scenario would result in an unreachable FTP server. By connecting these servers together into one

---

<sup>1</sup>See <http://www.cantrip.org/nobugs.html>

<sup>2</sup>It must be noted, however, that e.g. some versions of MSDE (limited free runtime of MS SQL) have been observed to require MS Office to install.

single server, Microsoft's design decision enabled a successful attack on FTP to bring down all other IIS services with it.

Wintel servers are always running some services that are not strictly necessary on a server. The Windows Graphical User Interface cannot be switched off, and although it is actually possible to remove Microsoft Networking, the system complains with a false warning that scares most less experienced administrators from doing it. Indexing server is often run behind IIS without being used.

For remote administration, Windows administrators often resort to telnet or VNC on the older variants of Windows, which means poorly authenticated and unencrypted connections are made to the servers. Since Windows 2000, Microsoft has bundled a Terminal Server into Windows which brought a major improvement, however it still lacks a file transfer capability which forces the administrators to complement it with FTP, SMB or other insecure file transfer services.

Windows services are often using Administrator or System account to run, which means that by compromising the server programs, an attacker usually gains enough rights to manipulate the whole server, or at least most of its services.

As SQL Slammer proved, many of the Wintel servers are running the same insecure service opened either to the Internet, or within large corporate networks. A virus or worm is thus able to count on an abundance of very similar targets and the effort required to endanger a large number of Wintel servers not significantly higher than an attack on a single server.

### 2.2.2 Other Platforms

Unix based servers are also suffering from little variety of the software they are running. Most of the Web servers are run on Apache (though there are a number of iPlanet, Roxen and others), a great majority of DNS servers run a version of bind. A large majority of Unix servers also depend on OpenSSH (and with that, on zlib and OpenSSL).

Many older Unix machines used to run sendmail for e-mail handling, but quite a lot of them have switched to more secure alternatives like Postfix or Qmail by now.

Fortunately, different CPU architectures and many differences in the Unix variants are making a universal attack difficult. Even those attacks that are successful may be limited by the good practice of running every service under a different UID which has only those rights that are necessary for the service to run. Some administrators also run publicly available services in a so-called chrooted environment,<sup>1</sup> which further isolates the service from the server.

In comparison with the Wintel world, attacks are limited in scope by several factors, but across a single hardware platform and one operating system, some attacks may endanger a rather significant number

---

<sup>1</sup>chroot is a command that allows an administrator to run a program with a different root directory. The program then has no access to files above or beside the chosen directory.

of servers as well, although by compromising one service, the attacker may not always be able to compromise anything else on the server.

## 2.3 The infrastructure

One of the most recent attacks, SQL Slammer deserves slightly more attention than others, because it impacted not only vulnerable systems and networks of negligent administrators, but practically everyone on the Internet—by using up all available bandwidth, the attack effectively became the first global DDoS, though this was probably only a side effect. One could argue that this DDoS was so successful because the Internet infrastructure in itself is a very homogeneous environment. From the protocols, all built on IPv4, to hardware which is mostly Cisco based, the Internet is just as vulnerable as any other homogeneous entity.

## 3 Learning from the Past

“The PC industry—and Microsoft—have done a terrific job of training us to live with shoddy and buggy products, something we do not tolerate in our cars, TVs, or phones.”

—Jai Singh, Editor of CNet’s NEWS.COM, July 1998

Although Melissa was the first attack so successful to become well known to general public, it was only one in a long line of attacks whose common denomination was that they were aided by homogeneity.

Starting with the legendary Morris worm, over Chernobyl, Love Bug, Explore.Zip to Nimda, Code Red, SQL Slammer and various other recent attacks, the level of homogeneity of the victim’s environment had always considerable impact on the damages.

It is nevertheless a sad fact that the reaction of the vendors was, and still seems to be, inadequate, or even counterproductive. For most software developers,<sup>1</sup> the notion of security is perceived as a nuisance, something that hinders and slows down development, prevents adding functions, makes interoperability difficult and, last but not least, has little impact on their salaries. From a developing company viewpoint, liability is mostly a non-issue thanks to carefully crafted disclaimers and licensing conditions. After all, users don’t expect software to be flawless and few of them would be able to distinguish mere bugs from design flaws anyway.

Therefore, security is often an afterthought, if it is considered at all during software development. It is often left out completely with reference to third party security devices, like anti-virus software, firewalls, intrusion detection etc.

---

<sup>1</sup>We should possibly exclude developers of so-called “trusted” products like e.g. Trusted Solaris, where security is taken extremely seriously.

Instead of concentrating on making their products secure, software developers have been thoughtlessly adding features, improving and re-organizing user interface, changing licensing schemes, renaming the products and release new versions prematurely, without adequate testing, as Marketing Departments are apparently taking over R&D. Many products, even security products, are poorly documented and/or emit all kinds of undocumented cryptic error messages, possibly in an attempt to coerce customers into buying expensive support packages. To make things worse, companies are often trying to cover up bugs, and do not release the information they have nor workarounds or bug fixes for the problems in a timely manner and at suitable fora. Some companies even engage in despicable new fashion of suing those who discover and publish bugs and flaws of their products, in what seems to be a desperate attempt to hide their own failures, further limit liability and hinder research and free speech in the scientific community.

As the largest software developer in the world, Microsoft is without doubt responsible for much of the problems that are now plaguing the Wintel platform. Although far from accepting the responsibility, after almost thirty years in business Microsoft finally recognized the need for security last year and gave their developers a one-month pause to learn about security and review their code for possible bugs. While it was certainly a big step forward in the right direction, it is not yet clear what effect it is going to have in the long term. One month is neither enough to review all the millions of lines of code, nor acquire the “security first” approach to development that is needed, especially for developers who must deal with all the design decisions inherited from the past.

## 4 Conclusions

“Microsoft’s biggest and most dangerous contribution to the software industry may be the degree to which it has lowered user expectations.”

—Esther Shindler, OS/2 Magazine

What allowed so many vendors to behave irresponsibly? Was it indeed lack of customer interest in security as suggested by Bill Gates in 1995? And if it is, shouldn’t vendors automatically have disregarded such lack of interest and automatically subordinated their development to requirements of security experts in their customers’ best interest? In any case, they have not, and now we are learning the consequences.

We should be also learning how to prevent or at least minimize the consequences, and how to secure more responsible behaviour of both vendors and customers for the future.

It seems that in the past, software developers lacked proper incentives to pay security enough attention. Following are suggestions that could gradually improve the situation:

1. Education of both Users and Administrators

- a) Regular user education on elementary rules how to use the Internet securely is necessary to reduce the impact of worms and viruses that may often rely on user misconduct.
- b) Administrators should be educated on security even if they are not primarily security administrators. They should follow both vendor and public security sources—both to further educate themselves on incidents and developments in the security field, and to be informed about possible problems in software in their care.
- c) Vendor decision makers and developers should be educated on security and learn to use the knowledge to avoid insecure designs.

All the groups must acknowledge the fact that education is a never ending process, as new kinds and methods of attacks are being developed all the time.

## 2. Preventing Homogeneity

- a) For every application, the most suitable platform should be selected, and slight preference should be given to platforms different from those that are currently in use. For example, if a company uses MS Exchange for their internal mail and groupware servers, it would be advisable that a Postfix on Linux or FreeBSD is used as the mail gateway to the Internet instead of another Microsoft Exchange. If a Webmail front end to the mailing system runs on MS IIS, other web servers like Apache might be a good choice for the company Internet presentation. And if Microsoft Windows are the operating system at most of the company desktops, it is advisable to use some Unix variant with Samba file sharing software as the file servers.
- b) Proprietary data formats should be avoided if possible, especially when data exchange with external subjects are planned, because using them may lock up users with one vendor, thus promote homogeneity and in case vendor's security record deteriorates, make transition to a different vendor difficult.
- c) Whenever a decision maker or project manager selects platform and/or a software package, he/she should check for vendor history to prevent future problems if vendor is not able to behave in a fair and responsible manner in case of security incidents or problems. Attention should be brought to recent security issues of the chosen product and an overview of vendor reactions should be made.

## 3. Legal Responsibility

Just as many other aspects of life, also software development must follow the same patterns of rights and responsibilities.

For commercially available software, there should be a legal framework that would override legal disclaimers most vendors are pushing onto the customers:

- a) First of all, commercial software, just as any other commercial products, should be bound by warranty rules of the country. All bugs and design flaws a vendor is aware of would have to be published and fixed, and the documented fixes should be available to the customers free of charge at least within the warranty period.
- b) Second, software vendors should be legally liable for damages caused by or allowed by bugs or design flaws in their software, except<sup>1</sup> in cases when they are behaving responsibly by some legal definition—e.g. that all bugs must be openly published as soon as one of the following conditions is met:
  - when a bugfix or workaround is available,
  - an exploit is or seems to be in the wild, or
  - the problem is over a month<sup>2</sup> old.
- c) Standard anti-monopoly regulations should be strictly applied to vendors with large market share to prevent possible misuse of monopoly power, boost competition and thus reduce homogeneity of used hardware and software.

The computer industry, both hardware manufacturing and software development, is still quite young, and it is developing extremely fast. However, over the last decade it started to influence everyday life of almost everybody, and thanks to the Internet, its products have spread beyond all expectations. Just as other industries in the past, also the computer industry shall have to adapt to the needs of non-experts, common people without the slightest understanding of things they are using.

Even though computer education becomes better every year, for most people it is quite distant and uninteresting, and most of the education concentrates on topics like “How to use Word to write a letter”. Therefore, the developers, administrators and decision makers will have to employ all their skills to make the computer environment not only user-friendly, but mainly as safe as possible for a common user. Preventing creation of fully homogeneous environments in which all the users can fall victims of a single attack should form an important part of the strategy. Users should be encouraged to choose their tools from a set of alternatives defined by administrators. Operating Systems (if needed at all on workstations, because thin clients are in many respects more secure for common users), Office Suites, Mail and Groupware clients, Web Browsers, chat clients etc. should depend on user choice, provided that the administrators are able to secure standards-based interoperability between the programs.

---

<sup>1</sup>The exception is necessary, as bugs are unfortunately inevitable and even though proper pre-release testing should discover at least the most grave errors, it is quite certain that at least some bugs always make it to the release. Similarly, although design flaws should be eliminated by thorough preparation before a project is started, there are many factors, like lack of experience, or information and discoveries not available at the time of the project beginning, that may ill-affect the design. However, once a design flaw is identified, it should be removed or worked around as soon as possible, and avoided in the future so that as little damage as possible is caused.

<sup>2</sup>It could be a different period, a fortnight, three weeks, however one month period has become quite an accepted community standard.

Heterogeneous environment may be more expensive from the administration and maintenance standpoint, however allowing growth of a monoculture, though cheaper and easier in the short run, is likely to have dangerous and expensive consequences later.