A Cost Analysis of Typical Computer Viruses and Defenses

by Fred Cohen †

Search Terms: Computer Viruses, Computer Risk Analysis

Abstract:

Various properties of computer viruses have been studied at length by many authors [1], but one of the areas where research results are relatively rare is evaluation of the costs of defenses. In this paper, we explore the costs of computer virus defenses in typical computing environments.

1 Background

The major computer virus defenses in widespread use today are "scanners" [2], "monitors" [3], "cryptographic checksums" [4], and "integrity shells" [5]. Other practical techniques that are viable for this purpose, but have not been put into widespread use as virus defenses are limited function [6,7], limited sharing [6,7], and sound change control [8].

Virus scanners are programs that search for known computer viruses. They normally work by searching some of the files on a disk for known and easily detected viruses. Some scanners also use special indicators to detect harder to identify viruses. For example, some viruses force the seconds field in the file modification time under DOS to 61 seconds, and some scanners use this as an indicator of that virus. Most virus scanners only cover viruses in binary executable files and only cover viruses that have fixed locations relative to the start of a program. Viruses with more complex infection mechanisms (e.g. random placement within a program), complex evolutionary viruses, and viruses that infect interpreted files (e.g. "Basic" viruses), are rarely scanned for by modern scanners. This is because these types of scans require complete examination of files, and thus a lot of time.

Cryptographic checksums are hard to forge many-to-one transforms that typically take a user supplied key and file, and produce a number [4]. By summing files with this technique and subsequently checking them on a regular basis, we can usually detect unauthorized changes, even when an attacker is trying to avoid detection.

Integrity shells are command interpreters that look for changes in interpreted information before interpreting it. They normally use cryptographic checksums for detection, can detect all primary infection and prevent all secondary infection, and are optimal for virus defense in untrusted systems [5]. Integrity shells can check all interpreted information, not just binary executable files.

Virus monitors are programs that look for known viruses each time an executable program is run. This is a special case of the integrity shell technique, where instead of using a cryptographic checksum to detect change, we look only for known viruses. Like scanners, monitors typically look only for viruses with simple infection mechanisms at known locations.

2 Notation and Assumptions

We will abbreviate as follows:

T_s	Total for scanner	s	systems
T_c	Total for crypto checksum	c	checks/year
T_m	Total for monitor	e	employee cost/min
T_i	Total for integrity shell	u	(d)(update-count)
t_s	minutes per scan	t_c	minutes per check
l_s	license for scanner	l_c	license crypto-checksum
l_m	license for monitor	l_i	license integrity shell
a_n	new attacks	a_o	old attacks
r_s	system cleanup costs	r_f	file cleanup costs
d	distribution costs	o_i	comm-rate$^{[K/c]}$

Most of these terms are self explanatory, but a few are a bit obscure. The licensing cost for crypto-checksums and integrity shells is normally a one-time cost, whereas for scanners and monitors, regular updates force licenses to be paid over time. To compensate, we use 10% per year of the total licensing cost for integrity shells and crypto-checksums as an equivalent to the yearly licensing cost for scanners and monitors. o_i is a term that describes the rate of spread of a virus, and has experientially been about 2 for a typical PC in a typical environment, and about 10 for a typical PC in a LAN environment. The update count is the number of times per year we do updates of scanners and monitors (which require updates to stay effective), and the distribution cost is the cost of each distribution (or redistribution) and installation of a software package.

We will also assume, for comparison purposes, that all performance is measured relative to a 4 Mhz PC-XT with 65 msec hard disk drives. Many current processors are far faster. For example, it is not unusual to see a 32 Mhz PC with an 18msec hard disk, which would be from 20 to 30 times faster than our assumed performance levels. Performance variations will also be discussed later, so we hope to cover this difference at that time.

Another important assumption we make is that the number of attacks per year does not vary with the number of systems. This is clearly not a valid assumption in many cases. To compensate for this, we discuss the effect of making the number of attacks linear in the number of systems.

3 Analysis of Virus Scanners

Most scanners and monitors are licensed on a per system per year basis. They require regular updates and must be used on a regular basis in order to be effective. They cover some viruses, and don't cover others, and there is a significant delay between the initial release of a virus and its inclusion in scanners and monitors. For most organizations, there is a substantial cost associated with the cleanup of a virus once detected. The more often we scan, the less a virus spreads before detection, but at the same time, scanning takes a significant amount of time, and that time can be quite costly when spent on a large number of systems on a regular basis.

The yearly operating cost of performing checks, assuming that employees are relatively idle while the checks are being done, is the number of checks per year times the employee cost times the time required for each scan. The update cost is the cost of distribution and installation times the number of updates per unit time. The license fee is set contractually. All of these are linear in the number of systems being scanned, so the yearly operating cost of performing scans is calculated as follows:

$$\text{yearly-operating-cost} = s[cet_s + l_s + u]$$

Typical figures are; one check each working day (usually at bootup), US$20 per hour (1/3 of a dollar per minute) for idle employee costs, 3 minutes per bootup scan, US$10 per year for licensing fees, US$5 per update, and 4 updates per year. Thus the yearly operating cost per system of scanning comes to:

$$((250)(1/3)(3)) + 10 + ((5)(4)) = 250 + 10 + 20 = \$280/\text{system/year}$$

The most striking thing here is that licensing costs are the least important factor in the cost of scanning. Even the update cost, which tends to exceed licensing cost, is trivial compared to the cost of performing daily checks. By using weekly checks instead of daily checks, or by scheduling coffee breaks during checking times, the cost of checking can be dramatically reduced. Unfortunately, scanning is usually done at system startup, and employees tend to sit and wait for the system to become ready at startup.

If there are viruses in the environment, the situation changes considerably. For viruses that are not detected by the current version of the scanner, it is a reasonable assumption that they will spread throughout all of the systems in the environment over a period of days to months, depending on the communications methods in use. We also assume that eventually, a cleanup will be necessary or desired, and for the moment, we will assume that there are no side effects of the virus such as data diddling, file deletion, etc. [9] Cleanup costs are assumed to include all down-time of equipment, employee time spent on cleanup, and time wasted while awaiting cleanup. Thus we calculate the "new attack cost" as follows:

$$\text{new-attack-cost} = a_n r_s s$$

It is hard to assess a typical environment today because of poor reporting, poor detection, and a general lack of statistical data on the subject. According to the IBM high integrity research laboratory [10], they don't find out about a virus until it has spread to about 10 major organizations. They were finding new viruses at a rate of more than one per week as of February, 1990. If a complete system cleanup costs $100, you would expect to spend $100 per system per year in cleanup costs from each new virus.

For known attacks that enter the environment, a regularly used scanning program can be quite effective in limiting damage. The spread of a virus in a computing environment is actually quite complex, and substantial work has been done to make an accurate model [11]. Another less accurate model that assumes an infinite population and homogeneous communications predicts (not surprisingly) exponential spread [12]. For simplicity, we will take the position that exponential spread occurs based on the communications rate, but is limited by the number of systems available for infection. Spread rate has been found to be 10 times larger in a networked environment than in an environment without networks [10], and clearly this should be taken into account. The old attack cost can then be calculated as:

$$\text{old-attack-cost} = a_o r_s \min[s, o_i]$$

In this model, as the checks per year goes down, the system tends towards increased cleanup costs. K can be calculated based on measured spread rates. Our observations have indicated that a virus in a non-networked environment spreads to 2 systems in the first day, 4 in the second day, etc. Thus, for a comm-rate of 2 for non-networked systems (20 for LANs), a check every business day, cleanup costs of $100 per system, 3 old viruses per year, and any number of systems over 2, we get a yearly cleanup cost as follows:

$$(3)(\$100)(\min[(>2), 2^{250/250}] = \$600$$

For a LAN environment with 20 old viruses per year, 20 or more systems, and all other factors being equal, we get:

$$(20)(\$100)\min[(>20), 20^{250/250}] = \$40{,}000$$

If we try to reduce checking costs by only checking once per week, we find that the expected loss from attacks goes up exponentially. For example, in the LAN environment above, we get:

$$(20)(\$100)\min[s, 20^{250/50}]$$
$$= \min[(\$2{,}000s), \$200{,}000{,}000]$$

Checking often is very important in terms of reduced cleanup costs when viruses appear, and this tends to dominate all other factors when checks are done infrequently.

The total cost, not including any side effects caused by viruses, is simply the sum of the costs we have given so far and a one-time initial distribution cost. The cost per system per year is then simply derived.

$$T_s = \text{yearly-operating-cost} + \text{new-attack-cost} + \text{old-attack-cost} + d$$
$$\text{cost-per-system-per-year} = T_s/s$$

There are some other time related factors in considering the future of virus scanners. For example, a typical scanner (circa 1/1/90) on a PC-XT, took 3 minutes per scan and reliably scanned for about 50 file viruses. As of 9/1/90, these figures were up to over 200 viruses, and scan times of about 15 minutes. If this trend continues, we will see 1 hour scan times within the next year or two! If virus designers continue to improve their techniques, as they have been doing lately, scanners will have to begin full searches in order to be effective against new strains of viruses. This will increase scanning times by at least an order of magnitude, and there is no reason to believe that other factors will compensate for this reduced performance.

4 Analysis of Virus Monitors

Virus monitors are essentially the same as virus scanners except that they exploit the integrity shell concept to eliminate most of the cleanup costs and dramatically increase scanning rates without dramatically increasing scanning costs. Since a scan of each program is performed before it is run, like the integrity shell, a monitor detects primary infections and prevents secondary infection by known viruses. Some state-of-the-art monitors even have special purpose routines for removal of many known viruses, thus eliminating most of the costs of repair.

Because checking a single file for a known virus is currently very fast, there is no real penalty for the checking time. Thus we get the modified form of the scanner equation.

$$\text{yearly-operating-cost} = s[l_m + u]$$

Using the same basis as was used for scanners above and a $5 per year licensing cost, the cost per system for using monitors comes to only:

$$5+(5)(4)=5+20=\$25/\text{system/year}$$

In this scenario, we can even afford to distribute new versions of the monitor every month and save money over the scanning program. At 12 distributions per year, we only get $65 per system per year, around 1/4 of the cost of scanning.

The cost of unmonitored attacks with a virus monitor are the same as the costs for a scanner, thus we get:

$$\text{new-attack-cost} = a_n r_s s$$

The cost for monitored attacks goes down considerably with a monitoring system because there is effectively no delay between the introduction of a virus and its detection. In essence, a known virus planted in a program is detected before it is ever used, and thus before it can

spread at all. The cleanup costs for monitored attacks is also very low, since it involves only the replacement of a single file.

$$\text{old-attack-cost}=a_o r_f$$

With an automated cleanup facility, this cost goes to very nearly zero, but since the cost is so low to start, it does not have a noticeable impact on overall attack costs unless the number of known attacks is very large and the number of unknown attacks is very small.

As in the case with scanners, monitors have total costs as follows:

$$T_m = \text{yearly-operating-cost} + \text{new-attack-cost} + \text{old-attack-cost} + d$$

$$\text{cost-per-system-per-year} = T_m/s$$

5 Analysis of Cryptographic Checksums

Cryptographic checksums can be used in a manner similar to scanners, in that they are periodically run to check for the presence of viruses, but there are significant differences between the two technologies.

The major disadvantage of cryptographic checksums is that they take longer than most current scanners because they have to perform analysis on the entire file instead of looking only in a few key places. Eventually, scanners will also have to do this to remain effective (even now, some viruses cannot be detected by scanners without a fill file scan), and when that time comes, cryptographic checksums will be faster because they perform simpler operations than the multiple string search operations required by full file scanners.

The major advantages of cryptographic checksums are that they detect (probabilistically) all modifications, and thus detect all viruses, whether previously known or previously unknown [4], and thus they don't require updates. Current high speed cryptographic checksums operate at a rate of about 20 Kbytes per second on a standard PC-XT running at 4MHz. A 20Mbyte disk thus takes about 1,000 seconds to verify, or about 10-15 minutes depending on the file structure complexity. This is only 3 times the time required for a scanner circa 1/1/90, and comparable to the time for the most thorough scanners circa 9/1/90.

The yearly operating costs can be calculated as shown below. Notice that there are no update costs, but that for the time being t_c is higher than t_s by a factor of 1 to 3.

$$\text{yearly-operating-cost} = s[cet_c + l_c]$$

There is a minor difficulty in calculating licensing costs because unlike scanners, checksum programs do not have to be periodically updated. As a result, they are usually purchased once, installed once, and paid for all at one time. To keep costs in yearly terms, we assume that the one-time cost of a checksum program is paid at a rate of 10% of the purchase cost

per year. Thus, the yearly licensing cost of a checksum program is taken to be 1/10 of the one-time licensing cost.

Typical figures for a large number of systems are \$50/system in licensing fees (\$5/system/year). We can operate at 3 minutes/check if only executable files are checked (which is comparable to the case for scanners and monitors). Everything else remaining the same, so we come to a typical cost of:

$$(250)(1/3)(3) + 5 = \$255/\text{system/year}$$

As in the case of scanners, the dominant cost is the cost of using this technique every day. The cost of all attacks under a cryptographic checksum system are the same as the cost of known attacks for virus scanners, while the total costs are the sum of attack costs and checking costs.

$$\text{attack-cost} = (a_o + a_n) r_s \min[s, o_i]$$
$$T_c = \text{attack-cost} + \text{yearly-operating-cost} + d$$

6 Analysis of Integrity Shells

Integrity shells are similar to cryptographic checksums in that they don't require updates, and similar to monitors in that they don't have daily checking overhead and are optimal in their detection of attacks, thus preventing viral spread. Thus the yearly operating cost for integrity shells comes to only the licensing fee.

As with checksumming systems, integrity shells do not require periodic updates, so we will again treat the yearly licensing cost as 10% of the one-time licensing fee.

Attack related costs with integrity shells are the same as monitored attack costs for virus monitors. As with monitors, state-of-the-art integrity shells completely automate the repair process for known and unknown corruptions in most cases, but there is an overhead associated with this repair (this will be covered in detail later)

Thus we have the total cost of integrity shells as:

$$T_i = (a_o + a_n) r_f + s l_i + d$$

In large quantities, integrity shells cost about \$50/system, which comes to a time deferred cost of \$5/system/year. Assuming that there are 3 known and 2 unknown attacks per year, and that the cost for a single file cleanup is \$10, the total yearly cost for integrity shells, including all costs of attack and defense, comes to only $\$[50 + 5s + 5]$! For an operation with 100 systems, this comes to a total of \$555, or only \$5.55/system/year.

7 Cost Comparisons

We start by restating the total costs from the four techniques under consideration:

$$T_s = s[cet_s + l_s + u] + a_n r_s s + a_o r_s \min[s, o_i] + d$$
$$T_c = s[cet_c + l_c] + [a_n + a_o] r_s \min[s, o_i] + d$$
$$T_m = s[l_m + u] + a_n r_s s + a_o r_f + d$$
$$T_i = s l_i + [a_n + a_o] r_f + d$$

The assumed values used for comparison above were:

c	250	e	1/3
t_s	3	t_c	3
l_s	5	l_c	5
l_m	5	l_i	5
a_n	2	a_o	3
r_s	100	r_f	10
d	5	o_i	2 (no LAN)
u	20	o_i	20 (LAN)

We will assume for comparison purposes that $o_i < s$ (i.e. we always detect viruses before all systems are infected), and thus simplify $\min[s, o_i]$ to o_i.

To make comparisons, we will subtract equations. We begin by comparing T_s to T_c as follows:

Term 1		Term 2
$T_s = s[cet_s + l_s + u]$	+	$a_n r_s s + a_o r_s o_i + d$
$T_c = s[cet_c + l_c]$	+	$[a_n + a_o] r_s o_i + d$
$T_s - T_c = s(ce[t_s - t_c] + [l_s - l_c + u])$	+	$a_n r_s [s - o_i]$

Unless checking is so rare that all systems get infected before checking detects an infection, $o_i << s$, and dividing by s, we get a per system cost difference of:

$$ce[t_s - t_c] + u + [l_s - l_c] + a_n r_s$$

With daily scanning or checking, any reasonable difference in licensing fees ($l_s - l_c$) are dominated by scanning and checking costs ($t_s - t_c$). Update costs (u) also tend to dominate licensing fee differences for this sort of system, so we can simplify in most cases to:

$$u - ce(t_c - t_s) + a_n r_s$$

so scanners are less expensive if and only if:

$$u + a_n r_s < ce(t_c - t_s)$$

In other words, if update costs and recovery costs from new attacks combine to exceed the checking cost difference between the two defenses, scanners are more expensive than cryptographic checksums. Using our previous numbers, we compute:

$$20+(2)(100) < (250)(1/3)(3\text{-}3)$$

As we see, if scanning and checksumming times are equal, scanners are always more expensive (assuming that licensing fee differences are not extraordinary). Put another way, if scanning and checksumming times are equal, the update costs and new virus eradication costs must be less than the licensing fee difference in order for scanners to be less expensive. For 4 updates per year at $5 per update, and assuming no unscanned for attacks, a checksumming system would have to cost $200 per system more than a scanner in order to be less cost effective ($200 at 10% per year is the same as $20 per year licensing fees). The only advantage of scanners for large numbers of systems is faster scan times, and this advantage will probably not last long.

We now compare scanners to monitors:

Term 1		Term 2
$T_s = s[cet_s + l_s + u]$	+	$a_n r_s s + a_o r_s o_i + d$
$T_m = s[l_m + u]$	+	$a_n r_s s + a_o r_f + d$
$T_s - T_m = s[cet_s + (l_s - l_m)]$	+	$a_o[r_s o_i - r_f]$

We assume that licensing fees for scanners and monitors are approximately the same relative to scanning costs ($[l_s - l_m] = 0$). We also note that $o_i > 1$ for any reasonable scenario, and the cost to clean a system is normally far greater than the cost to clean a single file ($r_s >> r_f$). We then get:

$$T_s - T_m = scet_s + a_o r_s$$

Since all of the terms are positive, T_s is always greater than T_m, and thus scanning costs always exceed monitoring costs.

We now quickly look at the relative costs of checksums and integrity shells:

Term 1		Term 2
$T_c = s[cet_c + l_c]$	+	$[a_n + a_o] r_s o_i + d$
$T_i = sl_i$	+	$[a_n + a_o] r_f + d$
$T_c - T_i = s[(l_c - l_i) + cet_c]$	+	$[a_n + a_o](r_s o_i - r_f)$

Assuming that licensing cost differences are not large, we can drop the $(l_c - l_i)$ term. Since $(o_i > 1)$ for any reasonable system, and $(r_s >> r_f)$ as we discussed earlier, we simplify to:

$$T_c - T_i = scet_c + [a_n + a_o] r_s o_i$$

Since all of these terms are positive, T_c is always greater than T_i, and thus integrity shells

are always more cost effective than cryptographic checksum programs.

Finally, we will compare integrity shells to monitors:

Term 1		Term 2
$T_m = s[l_m + u]$	+	$a_n r_s s + a_o r_f + d$
$T_i = s l_i$	+	$[a_n + a_o] r_f + d$
$T_m - T_i = s[(l_m - l_i) + u]$	+	$a_n r_s s + a_o r_f - [a_n + a_o] r_f$
$= s[(l_m - l_i) + u]$	+	$a_n [r_s s - r_f]$

The number of systems (s) is always at least 1, and as before, ($r_s >> r_f$). If we assume that yearly licensing cost differences ($l_m - l_i$) are small compared to update costs (u), we get a per system cost difference of:

$$T_m - T_s = u + a_n r_s$$

We conclude that integrity shells are always less expensive than virus monitors because all of these terms are positive. The cost difference comes from the added update cost required to keep monitors up to date (u), and the cost of cleanup for attacks that monitors don't detect ($a_n r_s$).

Another way to look at this, is that the yearly licensing cost of an integrity shell must exceed the yearly licensing cost of a monitor by the update cost and the undetected attack cleanup cost in order for a monitor to be more cost effective than an integrity shell.

$$l_i > l_m + u + a_n r_s$$

For $u = 20$, free monitor licensing, and assuming no unmonitored attacks ever take place, integrity shell license costs would have to exceed $200 per system ($20 per system per year) to be less cost effective than monitors.

8 Effects of Pre-Interpretation Checks

We assumed the time for monitors and integrity shells to check information at runtime to be negligible. The actual time (assuming PC-XT at 4Mhz, 62ms disk drive) is currently about 0.5 seconds for a typical monitor, and 20Kbytes per second for an integrity shell. The average DOS program is only about 17K bytes in length [13], so we would expect an integrity shell to take an average of 0.85 seconds, or about 0.35 seconds more than a monitor. Since a typical scanner takes 3 minutes and is used once per day, you would have to run about 211 programs per day, or one program every 2.5 minutes for integrity shell time costs to equal scanning costs. With monitors, this is one program every two minutes.

In typical secretarial environments, only 10 or 20 programs are run per day, and while programmers tend to run more programs, they tend to run smaller programs, and they

tend not to wait for them to finish before going onto the next task. Furthermore, most environments now have much faster machines, and the programs being run have significant startup times that cause the relative costs of monitors and integrity shells to be far less important. For example, on a 32Mhz PC with an 18msec disk drive, integrity shells take under 1/10 of a second per program, while the most thorough scanner still takes over 1 minute to check the executables at startup.

9 Effects of High Performance

Performance enhancements help cryptographic checksums and integrity shells a lot, but their impact on scanners and monitors is not as nice. As performance gets very high, scanning and monitor times are reduced, but periodic redistribution costs are not affected. We show the basic equations again:

$$\begin{array}{rcl} T_s = s[cet_s + l_s + u] & + & a_n r_s s + a_o r_s o_i + d \\ T_c = s[cet_c + l_c] & + & [a_n + a_o] r_s o_i + d \\ T_m = s[l_m + u] & + & a_n r_s s + a_o r_f + d \\ T_i = s l_i & + & [a_n + a_o] r_f + d \end{array}$$

For very fast systems, we assume that $t_s = 0$ and $t_c = 0$, and get:

$$\begin{array}{rcl} T_s = s[l_s + u] & + & a_n r_s s + a_o r_s o_i + d \\ T_m = s[l_m + u] & + & a_n r_s s + a_o r_f + d \\ T_c = s l_c & + & [a_n + a_o] r_s o_i + d \\ T_i = s l_i & + & [a_n + a_o] r_f + d \end{array}$$

If l_i, l_s, l_m, and l_c are about equal, and asusming that $r_f << r_s$ and $o_i << s$, we then find differences as follows:

$$\begin{aligned} T_s - T_m &= s[l_s - l_m] + a_o[r_s o_i - r_f] \\ &= 0 + a_o r_s o_i \end{aligned}$$

In other words, scanners and monitors differ only by repair costs from known viruses spreading to a relatively small number of systems (o_i).

$$T_m - T_c = su + a_n r_s s + a_o r_f - a_n r_s o_i - a_o r_s o_i$$

Since $s >> o_i$ and $o_i > 1$ and $r_f << r_s$, we get:

$$T_m - T_c = s[u + a_n r_s] - a_o r_s o_i$$

In this case, the only circumstance where $(T_m - T_c < 0)$, and thus where monitors are cheaper than checksums, is the case where:

$$s[u + a_n r_s] < a_o r_s o_i$$

In other words, in cases where update costs (u) and new virus cleanup costs ($a_n r_s$) are less expensive than cleaning old viruses spreading to a small number of systems ($a_o r_s o_i$), monitors are more cost effective. With large numbers of systems, in order for monitors to be less expensive than checksums, the number of old viruses would have to out number the number of new viruses by a factor equal to the number of systems. For an environment with 1,000 systems, that would mean 1,000 entries of old viruses would have to take place for every entry of a new virus in order for monitors to be more cost effective than cryptographic checksums. Clearly, as performance goes up, cryptographic checksums are more cost effective in large organizations.

Since performance is not an issue, we could theoretically check for viruses quite often without significant penalty. In the extreme, we could perform checksums every few seconds, and thus limit the spread of viruses to a very small portion of a system. In the limit, this is what an integrity shell does, except that it only checks those files which have to be checked when they have to be checked.

Integrity shells again come out on top as follows:

$$T_c - T_i = 0 + [a_n a_o][r_s o_i - r_f]$$

Since $r_s o_i >> r_f$, $T_c > T_i$, and integrity shells beat out cryptographic checksums. Our high performance assumption has no effect on $T_m - T_i$, except that it removes any minor advantage of monitors due to faster checking times at program invocation.

10 Effects of Online Backups and Automated Repair

Another cost related to monitors, scanners, integrity shells, and cryptographic checksums is the cost of automated repair, on-line backups, and manual repair with off-line backups. We begin with automated repair, which is essentially free (except for licensing costs) if done properly. To do this properly, we must make certain that we only cure known strains of known viruses. Only one product currently on the market is reliable in this way, but we assume that diligent effort will improve this situation with time.

On-line backups have real costs in terms of the disk space required to keep redundant information. In their simplest form, on-line backups double the disk space for all covered files. In a sense, the comparison here is unfair, because on-line backups can be used to cover all information, including data files, interpreted files, and other non-binary non-executable information. On-line backups can also repair unknown and known viruses as well as any other corruptions to stored information whether intentional or not. Thus the cost of coverage must be deferred over all corruptions for integrity shells and cryptographic checksums (which detect all corruptions), but only deferred over known viruses for monitors and scanners (since

they don't detect any other corruptions). This enhances the case for cryptographic checksums and integrity shells still further.

The cost of on-line backups is directly related to the cost of disk space. Current retail costs of disk space for personal computers runs in the range of $20/Mbyte for small disks, and $10/Mbyte for large disks. Most systems have relatively small disks, so let's use the $20 figure for the purpose of analysis. Assuming we are only covering binary executables with on-line backups, we only need to cover a small portion of the space on a typical system. A typical user PC has under 2Mbytes of its disk space taken by executable files (based on a very small survey of development systems). This would then come to a one-time cost of $40 per system for storage of on-line backups, or as we did with licensing fees, $4/year/system.

Unfortunately, this makes the assumption that you can buy disk space by the Mbyte, which you cannot. If you have extra disk space, on-line backups are essentially free, and if you don't, they are quite expensive. A rational approach might be to use on-line backups when there is available disk space, and use off-line backups when there is inadequate disk space.

Regardless of the technique applied for virus eradications, it has no significant impact on other virus related costs, since any eradication technique can be used with any detection mechanism. There is a tendency in the market, however, for integrity shells to use on-line backups, and for monitor to use customized cures. Clearly, we can mix these techniques as we desire, and perhaps an approptiate mix will give us the best of both worlds.

11 Effects of Cleaning Up Backups

We have also left out a major factor in the cleanup costs of monitors and scanners. If we are to do a thorough job of cleanup, we must not only remove viruses from on-line files, but also from backups. Otherwise, the backups will be infected, and when we try to restore from them, we will restore the viruses as well as the other information. With checksums and integrity shells, this is not a problem because no undetected changes occur, and thus backups don't become corrupted over a significant period of time, but with monitors and scanners, any time we get an update and detect a new virus, we must clean the entire history of backups in order to fully eradicate it.

Another closely related problem is that evolutionary viruses may release known strains into the environment. When we detect the known strain, we may be able to cure it, but we will not necessarily ever trace down its source, because it evolved from an unknown strain. This makes cleaning backups nearly impossible once an evolutionary virus of unknown origin infects the system over an extended period.

12 Cryptographic Checksum Maintenance

The only negative point we are aware of for cryptographic checksums and integrity shells, relates to the need to maintain the checksums as the files on the system change legitimately. These costs might dominate if cryptographic checksums and integrity shells had no provisions to deal with change. There are two factors that enhance the integrity shell's position in this arena.

One factor is that the binary executables covered by monitors and scanners change very rarely, while the non-executables not covered by monitors and scanners change more often. Thus, in order to be competitive with monitors and scanners, integrity shells must only be able to deal with the small rate of change in binary executables. In a typical system, binaries don't change more than once per year, so an automatic procedural method can often be used to maintain this mechanism.

As an example, in several organizations, the strategy has been adopted to issue an automatic resumming order whenever distributing an update to a binary executable. Thus along with the other installation routines on each widely distributed disk, comes a command to resum the new executables. This takes an extra few seconds during the installation process, but relative to other distribution costs, is trivial.

The second factor is that integrity shells on the market today have automation for handling change in a manner consistent with the organizational policy. Thus, if the organization determines that no changes are allowed without explicit user permission, this policy will be enforced. If changes are only allowed under specific conditions, those conditions can be explicitly and automatically enforced as well.

As an example, with a policy that no binary executable is to be trusted unless it has been explicitly checksummed during a distribution, no untrusted program (e.g. the AIDS disk of 1989) can be executed by the user. In the other extreme, a commonly used word processor changes its own binary executable to reflect user customizations, and may change itself at any time. This too is handled by integrity shells which can be told of this case and automatically resum this program after each execution, without leaving the window of vulnerability that would be present if we only resummed changed information on a periodic basis.

The net effect is that in a well controlled environment, controlling cryptographic checksums is a normal part of the change control process, and is essentially free after it is integrated into the environment.

13 Summary, Conclusions, and Further Work

We have provided a mathematical analysis of the costs of viruses and defenses for the most common defensive strategies. The analysis clearly shows that integrity shells are the most cost effective technique today in almost every case, and that their advantage over other techniques becomes larger as time passes, as performance increases, as the number of viruses increases, and as the number of systems being covered increases. Of the other techniques, there is a short period of time over which virus monitors will be more cost effective than cryptographic checksums in many cases, but as performance increases and the number of viruses increases, cryptographic checksums become better than monitors. Scanners, the most popular technique in the market at this time, are the least cost effective.

The analysis makes many assumptions about environments and viral spread, and some of them will be more or less valid in any given environment. This analysis is also not particularly deep, and it may well be that more thorough analysis will reveal more relevant information on the costs of viruses and defenses. Nevertheless, for a broad range of environments, the conclusions drawn seem to be valid.

Significant further work is called for in the area of mixed strategies for defense and the costs of other defensive methods. In particular, prevention through limited sharing and limited function should be explored for cost effectiveness, and mixes of all of these strategies should be considered.

References

1. F. Cohen, "A Summary of Recent Results on Computer Viruses", 1990 NIST/DOD Annual Conference on Computer Security.

2. F. Cohen, "A Note on the use of Pattern Matching in Computer Virus Detection", Invited Paper, Computer Security Conference, London, England, Oct 11-13, 1989, also appearing in DPMA Computer Virus Clinic, 1990.

3. J. Hirst, "Eliminator - Virus Detection and Removal", Users manual, British Computer Virus Research Centre, 1990

4. F. Cohen, "A Cryptographic Checksum for Integrity Protection", IFIP-TC11 "Computers and Security", V6#6 (Dec. 1987), pp 505-810.

5. F. Cohen, "Models of Practical Defenses Against Computer Viruses", IFIP-TC11, "Computers and Security", V7#6, December, 1988.

6. F. Cohen, "Computer Viruses", PhD Dissertation, University of Southern California, 1986, ASP Press (PO Box 81270, Pittsburgh, PA 15217 USA)

7. F. Cohen, "Computer Viruses - Theory and Experiments", DOD/NBS 7th Conference on Computer Security, originally appearing in IFIP-sec 84, also appearing in IFIP-TC11 "Computers and Security", V6(1987), pp22-35 and other publications in several languages

8. F. Cohen, "How To Do Sound Change Control and What It Costs", Information Protection, V1#6, June, 1990, ASP Press, (PO Box 81270, Pittsburgh PA 15217)

9. F. Cohen, "A Short Course on Computer Viruses", ASP Press, 1990 (PO Box 81270, Pgh PA 15217, USA)

10. S. White, "A Status Report on IBM Computer Virus Research", Italian Computer Virus Conference, 1990.

11. W. Gleissner, "A Mathematical Theory for the Spread of Computer Viruses", "Computers and Security", IFIP TC-11, V8#1, Jan. 1989 pp35-41.

12. Tipet, "The Tipet Theory of Computer Virus Propagation", Foundationware, USA.

13. M. Cohen, "A New Integrity Based Model for Limited Protection Against Computer Viruses", Masters Thesis, The Pennsylvania State University, College Park, PA 1988.