The text focuses on the use of laser-based lidar in oceanography.

The ability of lidar to penetrate into the ocean surface to obtain specific data in murky coastal waters is specially mentioned.

Particular attention is given to the advantage of laser-based lidars over passive satellite-based systems is obtaining signals not being contaminated by the water column or the bottom.

A typical lidar system is described with emphasis on the way it works.

This information may be of interest to research teams engaged in studying shallow waters.

Task II. Read the texts and write summaries according to the given one.

Text 1

Artificial Intelligence at Edinburgh University: a Perspective

Jim Howe

Revised June 2009.

Artificial Intelligence (AI) is an experimental science whose goal is to understand the nature of intelligent thought and action. This goal is shared with a number of longer established subjects such as Philosophy, Psychology and Neuroscience. The essential difference is that AI scientists are committed to computational modelling as a methodology for explicating the interpretative processes which underlie intelligent behaviour, that relate sensing of the environment to action in it. Early workers in the field saw the digital computer as the best device available to support the many cycles of hypothesizing, modelling, simulating and testing involved in research into these interpretative processes. They set about the task of developing a programming technology that would enable the use of digital computers as an experimental tool. Over the first four decades of AI's life, a considerable amount of time and effort was given over to the design and development of new special purpose list programming languages, tools and techniques. While the symbolic programming approach dominated at the outset, other approaches such as non-symbolic neural nets and genetic algorithms have featured strongly, reflecting the fact that computing is merely a means to an end, an experimental tool, albeit a vital one.

The popular view of intelligence is that it is associated with high level problem solving, i.e. people who can play chess, solve mathematical problems, make complex financial decisions, and so on, are regarded as intelligent. What we know now is that intelligence is like an iceberg. A small amount of processing activity relates to high level problem solving, that is the part that we can reason about and introspect, but much of it is devoted to our interaction with the physical environment. Here we are dealing with information from a range of senses, visual, auditory and tactile, and coupling sensing to action, including the use of language, in an appropriate reactive fashion which is not accessible to reasoning and introspection. Using the terms symbolic and sub-symbolic to distinguish these different processing regimes, in the early decades of our work in Edinburgh we subscribed heavily to the view that to make progress towards our goal we would need to understand the nature of the processing at both levels and the relationships between them. For example, some of our work focused primarily on symbolic level tasks, in particular, our work on automated reasoning, expert systems and planning and scheduling systems, some aspects of our work on natural language processing, and some aspects of machine vision, such as object recognition, whereas other work dealt primarily with tasks at the sub-symbolic level, including automated assembly of objects from parts, mobile robots, and machine vision for navigation.

Much of AI's accumulating know-how resulted from work at the symbolic level, modelling mechanisms for performing complex cognitive tasks in restricted domains, for example, diagnosing faults, extracting meaning from utterances, recognising objects in cluttered scenes. But this know-how had value beyond its contribution to the achievement of AI's scientific goal. It could be packaged and made available for use in the work place. This became apparent in the late 1970s and led to an upsurge of interest in applied AI. In the UK, the term Knowledge Based Systems (KBS) was coined for work which integrated AI know-how, methods and techniques with know-how, methods and techniques from other disciplines such as Computer Science and Engineering. This led to the construction of practical applications that replicated expert level decision making or human problem solving, making it more readily available to technical and professional staff in organisations. Today, AI/KBS technology has migrated into a plethora of products of industry and commerce, mostly unbeknown to the users.

History of AI at Edinburgh

The Department of Artificial Intelligence can trace its origins to a small research group established in a flat at 4 Hope Park Square in 1963 by Donald Michie, then Reader in Surgical Science. During the Second World War, through his membership of Max Newman's code-breaking group at Bletchley Park, Michie had been introduced to computing and had come to believe in the possibility of building machines that could think and learn. By the early 1960s, the time appeared to be ripe to embark on this endeavour. Looking back, there are four discernible periods in the development of AI at Edinburgh, each of roughly ten years' duration. The first covers the period from 1963 to the publication of the Lighthill Report by the Science Research Council in 1973. During this period, Artificial Intelligence was recognised by the University, first by establishing the Experimental Programming Unit in January 1965 with Michie as Director, and then by the creation of the Department of Machine Intelligence and Perception in October 1966. By then Michie had persuaded Richard Gregory and Christopher Longuet-Higgins, then at Cambridge University and planning to set up a brain research institute, to join forces with him at Edinburgh. Michie's prime interest lay in the elucidation of design principles for the construction of intelligent robots, whereas Gregory and Longuet-Higgins recognized that computational modelling of cognitive processes by machine might offer new theoretical insights into their nature. Indeed, Longuet-Higgins named his research group the Theoretical Section and Gregory called his the Bionics Research Laboratory. During this period there were remarkable achievements in a number of sub-areas of the discipline, including the development of new computational tools and techniques and their application to problems in such areas as assembly robotics and natural language. The POP-2 symbolic programming language which supported much subsequent UK research and teaching in AI was designed and developed by Robin Popplestone and Rod Burstall. It ran on a multi-access interactive computing system, only the second of its kind to be opened in the UK. By 1973, the research in robotics had produced the FREDDY II robot which was capable of assembling objects automatically from a heap of parts. Unfortunately, from the outset of their collaboration these scientific achievements were marred by significant intellectual disagreements about the nature and aims of research in AI and growing disharmony between the founding members of the Department. When Gregory resigned in 1970 to go to Bristol University, the University's reaction was to transform the Department into the School of Artificial Intelligence which was to be run by a Steering Committee. Its three research groups (Jim Howe had taken over responsibility for leading Gregory's group when he left) were given departmental status; the Bionics Research Laboratory's name was retained, whereas the Experimental Programming Unit became the Department of Machine Intelligence, and (much to the disgust of some local psychologists) the Theoretical Section was renamed the Theoretical Psychology Unit! At that time, the Faculty's Metamathematics Unit, which had been set up by Bernard Meltzer to pursue research in automated reasoning, joined the School as the Department of Computational Logic. Unfortunately, the high level of discord between the senior members of the School had become known to its main sponsors, the Science Research Council. Its reaction was to invite Sir James Lighthill to review the field. His report was published early in 1973. Although it supported AI research related to automation and to computer simulation of neurophysiological and psychological processes, it was highly critical of basic research in foundational areas such as robotics and language processing. Lighthill's report provoked a massive loss of confidence in AI by the academic establishment in the UK (and to a lesser extent in the US). It persisted for a decade - the so-called "AI Winter".

Since the new School structure had failed to reduce tensions between senior staff, the second ten year period began with an internal review of AI by a Committee appointed by the University Court. Under the chairmanship of Professor Norman Feather, it consulted widely, both inside and outside the University. Reporting in 1974, it recommended the retention of a research activity in AI but proposed significant organizational changes. The School structure was scrapped in favour of a single department, now named the Department of Artificial Intelligence; a separate unit, the Machine Intelligence Research Unit, was set up to accommodate Michie's work, and Longuet-Higgins opted to leave Edinburgh for Sussex University. The new Department's first head was Meltzer who retired in 1977 and was replaced by Howe who led it until 1996. Over the next decade, the Department's research was dominated by work on automated reasoning, cognitive modelling, children's learning and computation theory (until 1979 when Rod Burstall and Gordon Plotkin left to join the Theory Group in Computer Science). Some outstanding achievements included the design and development of the Edinburgh Prolog programming language by David Warren which strongly influenced the Japanese Government's Fifth Generation

Computing Project in the 1980s, Alan Bundy's demonstrations of the utility of meta-level reasoning to control the search for solutions to maths problems, and Howe's successful development of computer based learning environments for a range of primary and secondary school subjects, working with both normal and handicapped children.

Unlike its antecedents which only undertook teaching at Masters and Ph.D. levels, the new Department had committed itself to becoming more closely integrated with the other departments in the Faculty by contributing to undergraduate teaching as well. Its first course, AI2, a computational modelling course, was launched in 1974/75. This was followed by an introductory course, AI1, in 1978/79. By 1982, it was able to launch its first joint degree, Linguistics with Artificial Intelligence. There were no blueprints for these courses: in each case, the syllabuses had to be carved out of the body of research. It was during this period that the Department also agreed to join forces with the School of Epistemics, directed by Barry Richards, to help it introduce a Ph.D. programme in Cognitive Science. The Department provided financial support in the form of part-time seconded academic staff and studentship funding; it also provided access to its interactive computing facilities. From this modest beginning there emerged the Centre for Cognitive Science which was given departmental status by the University in 1985.

The third period of AI activity at Edinburgh begins with the launch of the Alvey Programme in advanced information technology in 1983. Thanks to the increasing number of successful applications of AI technology to practical tasks, in particular expert systems, the negative impact of the Lighthill Report had dissipated. Now, AI was seen as a key information technology to be fostered through collaborative projects between UK companies and UK universities. The effects on the Department were significant. By taking full advantage of various funding initiatives provoked by the Alvey programme, its academic staff complement increased rapidly from 4 to 15. The accompanying growth in research activity was focused in four areas, Intelligent Robotics, Knowledge Based Systems, Mathematical Reasoning and Natural Language Processing. During the period, the Intelligent Robotics Group undertook collaborative projects in automated assembly, unmanned vehicles and machine vision. It proposed a novel hybrid architecture for the hierarchical control of reactive robotic devices, and applied it successfully to industrial assembly tasks using a low cost manipulator. In vision, work focused on 3-D geometric object representation, including methods for extracting such information from range data. Achievements included a working range sensor and range data segmentation package. Research in Knowledge Based Systems included design support systems, intelligent front ends and learning environment. The Edinburgh Designer System, a design support environment for mechanical engineers started under Alvey funding, was successfully generalised to small molecule drug design. The Mathematical Reasoning Group prosecuted its research into the design of powerful inference techniques, in particular the development of proof plans for describing and guiding inductive proofs, with applications to problems of program verification, synthesis and transformation, as well as in areas outside Mathematics such as computer configuration and playing bridge. Research in Natural Language Processing spanned projects in the sub-areas of natural language interpretation and generation. Collaborative projects included the implementation of an English language front end to an intelligent planning system, an investigation of the use of language generation techniques in hypertext-based documentation systems to produce output tailored to the user's skills and working context, and exploration of semi-automated editorial assistance such as massaging a text into house style.

In 1984, the Department combined forces with the Department of Lingistics and the Centre for Cognitive Science to launch the Centre for Speech Technology Research, under the directorship of John Laver. Major funding over a five year period was provided by the Alvey Programme to support a project demonstrating real-time continuous speech recognition.

By 1989, the University's reputation for research excellence in natural language computation and cognition enabled it to secure in collaboration with a number of other universities one of the major Research Centres which became available at that time, namely the Human Communication Research Centre which was sponsored by ESRC. During this third decade, the UGC/UFC started the process of assessing research quality. In 1989, and again in 1992, the Department shared a "5" rating with the other departments making up the University's Computing Science unit of assessment.

The Department's postgraduate teaching also expanded rapidly. A masters degree in Knowledge Based Systems, which offered specialist themes in Foundations of AI, Expert Systems, Intelligent Robotics and Natural Language Processing, was established in 1983, and for many years was the largest of the Faculty's taught postgraduate courses with 40-50 graduates annually. Many of the Department's complement of about 60 Ph.D. students were drawn from its ranks. At undergraduate level, the most significant development was the launch, in 1987/88, of the joint degree in Artificial Intelligence and Computer Science, with support from the UFC's Engineering and Technology initiative. Subsequently, the modular structure of the course material enabled the introduction of joint degrees in AI and Mathematics and AI and Psychology. At that time, the Department also shared an "Excellent" rating awarded by the SHEFC's quality assessment exercise for its teaching provision in the area of Computer Studies.

The start of the fourth decade of AI activity coincided with the publication in 1993 of "Realising our Potential", the Government's new strategy for harnessing the strengths of science and engineering to the wealth creation process. For many departments across the UK, the transfer of technology from academia to industry and commerce was uncharted territory. However, from a relatively early stage in the development of AI at Edinburgh, there was strong interest in putting AI technology to work outside the laboratory. With financial banking from ICFC, in 1969 Michie and Howe had established a small company, called Conversational Software Ltd (CSL), to develop and market the POP-2 symbolic programming language. Probably the first AI spin-off company in the world, CSL's POP-2 systems supported work in UK industry and academia for a decade or more, long after it ceased to trade. As is so often the case with small companies, the development costs had outstripped market demand. The next exercise in technology transfer was a more modest affair, and was concerned with broadcasting some of the computing tools developed for the Department's work with schoolchildren. In 1981 a small firm, Jessop Microelectronics, was licensed to

manufacture and sell the Edinburgh Turtle, a small motorised cart that could be moved around under program control leaving a trace of its path. An excellent tool for introducing programming, spatial and mathematical concepts to young children, over 1000 were sold to UK schools (including 100 supplied to special schools under a DTI initiative). At the same time, with support from Research Machines, Peter Ross and Ken Johnson re-implemented the children's programming language, LOGO, on Research Machines microcomputers. Called RM Logo, for a decade or more it was supplied to educational establishments throughout the UK by Research Machines.
As commercial interest in IT in the early 1980s exploded into life, the Department was bombarded by requests from UK companies for various kinds of technical assistance. For a variety of reasons, not least the Department's modest size at that time, the most effective way of providing this was to set up a separate non-profit making organisation to support applications oriented R&D. In July 1983, with the agreement of the University Court, Howe launched the Artificial Intelligence Applications Institute. At the end of its first year of operations, Austin Tate succeeded Howe as Director. Its mission was to help its clients acquire know-how and skills in the construction and application of knowledge based systems technology, enabling them to support their own product or service developments and so gain a competitive edge. In practice, the Institute was a technology transfer experiment: there was no blueprint, no model to specify how the transfer of AI technology could best be achieved. So, much time and effort was given over to conceiving, developing and testing a variety of mechanisms through which knowledge and skills could be imparted to clients. A ten year snapshot of its activities revealed that it employed about twenty technical staff; it had an annual turnover just short of £1M, and it had broken even financially from the outset. Overseas, it had major clients in Japan and the US. Its work focused on three sub-areas of knowledge-based systems, planning and scheduling systems, decision support systems and information systems.

Formally, the Department of Artificial Intelligence disappeared in 1998 when the University conflated the three departments, Artificial Intelligence, Cognitive Science and Computer Science, to form the new School of Informatics.

Text 2
A gift of tongues

Troy Dreier
PC MAGAZINE July 2009.

1. Jokes about the uselessness of machine translation abound. The Central Intelligence Agency was said to have spent millions trying to program computers to translate Russian into English. The best it managed to do, so the tale goes, was to turn the Famous-Russian saying "The spirit is willing but the flesh is weak" into "The vodka is good but the meat is rotten." Sadly, this story is a myth. But machine translation has certainly produced its share of howlers. Since its earliest days, the subject has suffered from exaggerated claims and impossible expectations.

2. Hype still exists. But Japanese researchers, perhaps spurred on by the linguistic barrier that often seems to separate their country's scientists and technicians from those in the rest of the world, have made great strides towards the goal of reliable machine translation—and now their efforts are being imitated in the West.

3. Until recently, the main commercial users of translation programs have been big Japanese manufacturers. They rely on machine translation to produce the initial drafts of their English manuals and sales material. (This may help to explain the bafflement many western consumers feel as they leaf through the instructions for their video recorders.) The most popular program for doing this is e-j bank, which was designed by Nobuaki Kamejima, a reclusive software wizard at AI Laboratories in Tokyo. Now, however, a bigger market beckons. The explosion of foreign languages (especially Japanese and German) on the Internet is turning machine translation into a mainstream business. The fraction of web sites posted in English has fallen from 98% to 82% over the past three years, and the trend is still downwards. Consumer software, some of it written by non-Japanese software houses, is now becoming available to interpret this electronic Babel to those who cannot read it.

Enigma variations

4. Machines for translating from one language to another were first talked about in the 1930s. Nothing much happened, however, until 1940 when an American mathematician called Warren Weaver became intrigued with the way the British had used their pioneering Colossus computer to crack the military codes produced by Germany's Enigma encryption machines. In a memo to his employer, the Rockefeller Foundation, Weaver wrote: "I have a text in front of me which is written in Russian but I am going to pretend that it is really written in English and that it has been coded in some strange symbols. All I need to do is to strip off the code in order to retrieve the information contained in the text."

5. The earliest "translation engines" were all based on this direct, so-called "transformer", approach. Input sentences of the source language were transformed directly into output sentences of the target language, using a simple form of parsing. The parser did a rough/analysis of the source sentence, dividing it into subject, object, verb, etc. Source words were then replaced by target words selected from a dictionary, and their order rearranged so as to comply with the rules of the target language.

6. It sounds simple, but it wasn't. The problem with Weaver's approach was summarized succinctly by Yehoshua Bar-Hillel, a linguist and philosopher who wondered what kind of sense a machine would make of the sentence "The pen is in the box" (the writing instrument is in the container) and the sentence "The box is in the pen" (the container is in the[play]pen).

7. Humans resolve such ambiguities in one of two ways. Either they note the context of the preceding sentences or they infer the meaning in isolation by knowing certain rules about the real world—in this case, that boxes are bigger than pens (writing instruments) but smaller than pens (play-pens) and that bigger objects cannot fit inside smaller ones. The computers available to Weaver and his immediate successors could not possibly have managed that.

8. But modern computers, which have more processing power arid more memory, can. Their translation engines are able to adopt a less direct approach, using what is called "linguistic knowledge". It is this that has allowed Mr. Kamejima to produce e-j bank, and has also permitted NeocorTech of San Diego to come up with Tsunami and Typhoon - the first Japanese-language-translation software to run on the standard (English) version of Microsoft Windows.

9. Linguistic-knowledge translators have two sets of grammatical rules—one for the source language and one for the target. They also have a lot of information about the idiomatic differences between the languages, to stop them making silly mistakes.

10. The first set of grammatical rules is used by the parser to analyze an input sentence ("I read" The Economist "every week"). The sentence is resolved into a tree that describes the structural relationship between the sentence's components ("I" [subject], "read" (verb), "The Economist" (object) and "every week" [phrase modifying the verb). Thus far, the process is like that of a Weaver-style transformer engine. But then things get more complex. Instead of working to a pre-arranged formula, a generator (i.e., a parser in reverse) is brought into play to create a sentence structure in the target language. It does so using a dictionary and a comparative grammar—a set of rules that describes the difference between each sentence component in the source language and its counterpart in the target language. Thus a bridge to the second language is built on deep structural foundations.

11. Apart from being much more accurate, such linguistic-knowledge engines should, in theory, be reversible—you should be able to work backwards from the target language to the source language. In practice, there are a few catches which prevent this from happening as well as it might - but the architecture does at least make life easier for software designers trying to produce matching pairs of programs. Tsunami (English to Japanese) and Typhoon Japanese to English), for instance, share much of their underlying programming code.

12. Having been designed from the start for use on a personal computer rather than a powerful workstation or even a mainframe, Tsunami and Typhoon use memory extremely efficiently. As a result, they are blindingly fast on the latest PCs—translating either way at speeds of more than 300,000 words an hour. Do they produce perfect translations at the click of a mouse? Not by a long shot. But they do come up with surprisingly good first drafts for expert translators to get their teeth into. One mistake that the early researchers made was to imagine that nothing less than flawless, fully automated machine translation would suffice. With more realistic expectations, machine translation is, at last, beginning to thrive.

Text 3

IBM promises science 500-fold break-through in supercomputing power

David Stone

PC MAGAZINE March 8, 2009.

Biologists hail SI 00 million project to build a "petaflop" computer as likely to revolutionize our understanding of cellular biology. The computer, nicknamed 'Blue Genes', world be around 500 times faster than today's most powerful supercomputer. Computer scientists say that the planned machine, details of which were revealed last: week, is the first large leap in computer architecture in decades.

IBM will build the programme around the challenge of modeling protein folding (see below), with much of the research costs going on designing software. It will involve 50 scientists from IBM Research's Deep Computing Institute and Computational Biology Group, and unnamed outside academics.

But Blue Gene's hardware will not he customized to the problem and, if IBM's blueprint works, it will offer all scientific disciplines petaflop computers. These will be capable of more than one quadrillion floating point operations ('flop') per second - around two million times more powerful than today's top desktops. Most experts have" predicted that fundamental technological difficulties would prevent a petaflop computer being built before around 2015.

"It is, fantastic that IBM is doing this," says George Lake, a scientist at the university of Washington and NASA project, scientist for high-performance computing in Earth and space science. IBM is showing leadership by ushering in a new generation of supercomputers, he says.

The biggest-technological constraints to building a petaflop machine have been latency - increasing the speed with which a chip addresses the memory - and reducing power-consumption. A petaflop computer build using conventional chips would consume almost one billion watts of power. IBM reckons Blue Gene will use just one million-watts.

Although processor speeds have increased exponentially, the time to fetch *dm from* the memory of a supercomputer, 300 nanoseconds, is only slightly less than half what it was 20 years ago. Putting more and more transistors on a chip is therefore unlikely to lead to much greater speed.

"We set out from scratch, completely ignoring history, and thought how can we get the highest performance out of silicon," says Monty Denneau, a scientist at IBM's Thomas J. Watson research center in Yorktown Heights, New York, who is assistant architect of Slue Gene.

Arvind, a professor of computer science at Mit who is considered one of the top authorities on computer architecture, applauds IBM's approach. "It has made very big steps in rethinking computer architecture to try to do without the components that consume power, it has taken all these research ideas and pulled them together."

Task III. Write précis of the following articles.

Text 1

Antiviruses. Principle of work. Examples of antiviruses.

Antivirus software consists of computer programs that attempt to identify, thwart and eliminate computer viruses and other malicious software (malware). Antivirus software typically uses two different techniques to accomplish this:

• Examining (scanning) files to look for known viruses matching definitions in a virus dictionary
• Identifying suspicious behavior from any computer program which might indicate infection. Such analysis may include data captures, port monitoring and other methods.

Most commercial antivirus software uses both of these approaches, with an emphasis on the virus dictionary approach.

Historically, the term antivirus has also been used for computer viruses that spread and combated malicious viruses. This was common on the Amiga computer platform.

Dictionary

In the virus dictionary approach, when the antivirus software looks at a file, it refers to a dictionary of known viruses that the authors of the antivirus software have identified. If a piece of code in the file matches any virus identified in the dictionary, then the antivirus software can take one of the following actions:
• attempt to repair the file by removing the virus itself from the file
• quarantine the file (such that the file remains inaccessible to other programs and its virus can no longer spread)
• delete the infected file

To achieve consistent success in the medium and long term, the virus dictionary approach requires periodic (generally online) downloads of updated virus dictionary entries. As civically minded and technically inclined users identify new viruses "in the wild", they can send their infected files to the authors of antivirus software, who then include information about the new viruses in their dictionaries.

Dictionary-based antivirus software typically examines files when the computer's operating system creates, opens, closes or e-mails them. In this way it can detect a known virus immediately upon receipt. Note too that a System Administrator can typically schedule the antivirus software to examine (scan) all files on the computer's hard disk on a regular basis. Although the dictionary approach can effectively contain virus outbreaks in the right circumstances, virus authors have tried to stay a step ahead of such software by writing "oligomorphic", "polymorphic" and more recently "metamorphic" viruses, which encrypt parts of themselves or otherwise modify themselves as a method of disguise, so as not to match the virus's signature in the dictionary.

Suspicious behavior

The suspicious behavior approach, by contrast, doesn't attempt to identify known viruses, but instead monitors the behavior of all programs. If one program tries to write data to an executable program, for example, the antivirus software can flag this suspicious behavior, alert a user and ask what to do.

Unlike the dictionary approach, the suspicious behavior approach therefore provides protection against brand-new viruses that do not yet exist in any virus dictionaries. However, it can also sound a large number of false positives, and users probably become desensitized to all the warnings. If the user clicks "Accept" on every such warning, then the antivirus software obviously gives no benefit to that user. This problem has worsened since 1997, since many more nonmalicious program designs came to modify other **.exe** files without regard to this false positive issue. Thus, most modern antivirus software uses this technique less and less.

Other approaches

Some antivirus-software uses of other types of heuristic analysis. For example, it could try to emulate the beginning of the code of each new executable that the system invokes before transferring control to that executable. If the program seems to use self-modifying code or otherwise appears as a virus (if it immediately tries to find other executables, for example), one could assume that a virus has infected the executable. However, this method could result in a lot of false positives. Yet another detection method involves using a sandbox. A sandbox emulates the operating system and runs the executable in this simulation. After the program has terminated, software analyzes the sandbox for any changes which might indicate a virus. Because of performance issues, this type of detection normally only takes place during on-demand scans. Also this method may fail as virus can be nondeterministic and result in different actions or no actions at all done then run - so it will be impossible to detect it from one run. Some virus scanners can also warn a user if a file is likely to contain a virus based on the file type.

An emerging technique to deal with malware in general is whitelisting. Rather than looking for only known bad software, this technique prevents execution of all computer code except that which has been previously identified as trustworthy by the system administrator. By following this default deny approach, the limitations inherent in keeping virus signatures up to date are avoided. Additionally, computer applications that are unwanted by the system administrator are prevented from executing since they are not on the whitelist. Since modem enterprise organizations have large quantities of trusted applications, the limitations of adopting this technique rest with the system administrators' ability to properly inventory and maintain the whitelist of trusted applications. As such, viable implementations of this technique include tools for automating the inventory and whitelist maintenance processes.

Issues of concern

- The spread of viruses using e-mail as their infection vector could be inhibited far more inexpensively and effectively, without the need to install additional antivirus software; if bugs in e-mail clients, which allow the unauthorized execution of code, were fixed
- User education can effectively supplement antivirus software. Simply training users in safe computing practices (such as not downloading and executing unknown programs from the Internet) would slow the spread of viruses and obviate the need of much antivirus software.

- The ongoing writing and spreading of viruses and of panic about them gives the vendors of commercial antivirus software a financial interest in the ongoing existence of viruses. Some theorize that antivirus companies have financial ties to virus writers, to generate their own market, though there is currently no evidence for this.
- Some antivirus software can considerably reduce performance. Users may disable the antivirus protection to overcome the performance loss, thus increasing the risk of infection. For maximum protection the antivirus software needs to be enabled all the time — often at the cost of slower performance (see also software bloat).
- It is sometimes necessary to temporarily disable virus protection when installing major updates such as Windows Service Packs or updating graphics card drivers. Having antivirus protection running at the same time as installing a major update may prevent the update installing properly or at all.
- When purchasing antivirus software, the agreement may include a clause that your subscription will be automatically renewed, and your credit card automatically billed at the renewal time without your approval. For example, McAfee requires one to unsubscribe at least 60 days before the expiration of the present subscription, yet it does not provide phone access nor a way to unsubscribe directly through their website. In that case, the subscriber's recourse is to contest the charges with the credit card issuer.

History

There are competing claims for the innovator of the first antivirus product. Perhaps the first publicly known neutralization of a wild PC virus was performed by European Bemt Fix (also Bemd) in early 1987. Fix neutralized an infection of the Vienna virus. Following Vienna a number of highly successful viruses appeared including Ping Pong, Lehigh, and Suriv-3 aka Jemsalem. In January 1988, researchers in the Hebrew University developed "unvirus" and "immune", which tell users whether their disks have been infected and applies an antidote to those that have.

From 1988 onwards many companies formed with a focus on the new field of antivirus technology. One of the first breakthroughs in antivirus technology occurred in March 1988 with the release of the Den Zuk viruses created by Denny Yanuar Ramdhani of Indonesia. Den Zuk neutralized the Brain virus. April 1988 saw the Virus-L forum on Usenet created, and mid 1988 saw the development by Peter Tippett of a heuristic scanner capable of detecting viruses and Trojans which was given a small public release. Fall 1988 also saw antivirus software Dr. Solomon's Anti-Virus Toolkit released by Briton Alan Solomon. By December 1990 the market had matured to the point of nineteen separate antivirus products being on sale including Norton AntiVirus and ViruScan from McAfee.

Tippett made a number of contributions to the budding field of virus detection. He was an emergency room doctor who also ran a computer software company. He had read an article about the Lehigh virus were the first viruses to be developed, but it was Lehigh that Tippett read about and he questioned whether they would have similar characteristics to viruses that attack humans. From an epidemiological viewpoint, he was able to determine how these viruses were affecting systems within the computer (the boot-sector was affected by the Brain virus, the .com files were affected by the Lehigh virus, and both .com and .exe files were affected by the Jemsalem virus). Tippett's company Certus International Corp. then began to create anti-virus software programs. The company was sold in 1992 to Symantec Corp, and Tippett went to work for them, incorporating the software he had developed into Symantec's product, Norton AntiVirus.

Best antivirus soft

NOD32 is an antivirus package made by the Slovak company Eset. Versions are available for Microsoft Windows, Linux, FreeBSD and other platforms. Remote administration tools for multiuser installations are also available at extra cost. NOD32 Enterprise Edition consists of NOD32 AntiVirus and NOD32 Remote Administrator. The NOD32 Remote Administrator program allows a network administrator to monitor anti-virus functions, push installations and upgrades to unprotected PCs on the network and update configuration files from a central location.

NOD32 is certified by ICSA Labs. It has been tested 44 times by Virus Bulletin and has failed only 3 times, the lowest failure rate in their tests. At CNET.com, it received a review of 7.3/10.

Technical information

NOD32 consists of an on-demand scanner and four different real-time monitors. The on-demand scanner (somewhat confusingly referred to as NOD32) can be invoked by the scheduler or by the user. Each real-time monitor covers a different virus entry point:
AMON (Antivirus MONitor) - scans files as they are accessed by the system, preventing a virus from executing on the system.
DMON (Document MONitor) - scans Microsoft Office documents and files for macro viruses as they are opened and saved by Office applications.
IMON (Internet MONitor) - intercepts traffic on common protocols such as POPS and HTTP to detect and intercept viruses before they are saved to disc.
XMON (MS eXchange MONitor) - scans incoming and outgoing mail when NODS 2 is running and licensed for Microsoft Exchange Server – i.e, running on a server environment. This module is not present on workstations at all.

NOD32 Virus Detection Alert

NOD32 is written largely in assembly code, which contributes to its low use of system resources and high scanning speed, meaning that NOD32 can easily process more than 23MB per second while scanning on a modest P4 based PC and on average, with all real-time modules active, uses less than 20MB of memory in total but the physical

RAM used by NOD32 is often just a third of that. According to a 2005 Virus Bulletin test, NOD32 performs scans two to five times faster than other antivirus competitors. In a networked environment NOD32 clients can update from a central "mirror server" on the network, reducing bandwidth usage since new definitions need only be downloaded once by the mirror server as opposed to once for each client. NOD32's scan engine uses heuristic detection (which Eset calls "ThreatSense") in addition to signature files to provide better protection against newly released viruses.

Text 2
What is a virus?

B. Kelley
IOWA STATE UNIVERSITY, PM 1789 Rewised June, 2010.

In 1983, researcher Fred Cohen defined a computer virus as "a program that can 'infect' other programs by modifying them to include a ... version of itself. " This means that viruses copy themselves, usually by encryption or by mutating slightly each time they copy.

There are several types of viruses, but the ones that are the most dangerous are designed to corrupt your computer or software programs. Viruses can range from an irritating message flashing on your computer screen to eliminating data on your hard drive. Viruses often use your computer's internal clock as a trigger. Some of the most popular dates used are Friday the 13th and famous birthdays. It is important to remember that viruses are dangerous only if you execute (start) an infected program. There are three main kinds of viruses*. Each kind is based on the way the virus spreads.

1. Boot Sector Viruses - These viruses attach themselves to floppy disks and then copy themselves into the boot sector of your hard drive. (The boot sector is the set of instructions your computer uses when it starts up.) When you start your computer (or reboot it) your hard drive gets infected. You can get boot sector viruses only from an infected floppy disk. You cannot get one from sharing files or executing programs. This type of virus is becoming less common because today's computers do not require a boot disk to start, but they can still be found on disks that contain other types of files. One of the most common boot sector viruses is called "Monkey," also known as "Stoned."

2. Program Viruses - These viruses (also known as traditional file viruses) attach themselves to programs' executable files. Usually a program virus will attach to an .exe or .com file. However, they can infect any file that your computer runs when it launches a program (including .sys, .dll, and others). When you start a program that contains a virus, the virus usually loads into your computer's Memory.

*** Three kinds of viruses**

1. Boot Sector viruses attach to floppy disks and then copy into the boot sector of your hard drive.

2. Program viruses attach to a program's executable files.

3. Macro viruses attach to templates.

The truth about viruses

The majority of people believe that the most common source of viruses is the Internet through e-mail or downloaded files. The truth is, however, that the majority of viruses spread through shared floppy disks or shared files on internal network.

Even if you are not connected to the Internet you should still be concerned about viruses. You should also be aware that there are thousands of false rumors of viruses (virus hoaxes).