USENIX
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

The following paper was originally published in the

Proceedings of the Workshop on Intrusion Detection and Network Monitoring

Santa Clara, California, USA, April 9–12, 1999

The Packet Vault: Secure Storage of Network Data

C. J. Antonelli, M. Undy, and P. Honeyman
University of Michigan

For more information about the USENIX Association:
Phone: 1 510 528 8649 FAX: 1 510 548 5738
Email: office@usenix.org WWW: http://www.usenix.org

The Packet Vault: Secure Storage of Network Data

C.J. Antonelli
M. Undy
P. Honeyman

Center for Information Technology Integration
The University of Michigan
Ann Arbor
{cja,mundy,honey}@citi.umich.edu

Abstract

This paper describes the packet vault, a cryptographically secured archiver of network packet data. The vault captures network packets, encrypts them, and writes them to long-term CD-ROM storage for later analysis and for evidentiary purposes. The cryptographic organization of the vault permits selected traffic to be made available without exposing other traffic.

1. Introduction

The goal of the packet vault project at the Center for Information Technology Integration is to provide cryptographically secured long-term storage of network packets for later use as input data for intrusion detection algorithms or for possible evidentiary purposes.

Creating a complete, permanent record of all activity on a subnet addresses security threats in several ways. First, intrusion detection algorithms can be executed using the record as the input packet source. Detectors can be run over the same record with different parameter settings, outputs of different detectors can be compared, and new detectors can be created and evaluated against the record. Conducting such experiments requires a complete record of packets.

Second, in response to an intrusion in progress, the packet vault can be attached to a subnet under attack; the packets it stores may be used to determine quickly the source and nature of the intrusion, and thus help shape the response. In addition, the vault can be permanently connected to a suspect subnet, allowing the record to be examined periodically.

Finally, a properly constructed corpus of packet data may serve as evidence in legal proceedings.

In the remainder of this paper, we describe the goals of the packet vault, then discuss the hardware, software, and cryptographic organization of the vault. We then describe our experiences, discuss some issues — including legal issues and the strength of DES — in operating the vault, and conclude with a discussion of future work.

2. Goals

The architecture of the packet vault reflects the following goals:

- **Commodity.** We want to build a packet vault from commodity hardware and software, notwithstanding the attraction of expensive machines with fast buses and I/O devices. With a vault built from cheap parts in hand, we feel we can trade money for speed by buying faster parts (in a year).
- **Completeness.** To create a complete record, it is vital to capture and store every packet. We suspect that an adversary can exploit any form of packet triage; the only way to defend against all such attacks is to build a vault that stores packets at the maximum rate the network delivers them.
- **Permanency.** We decided from the outset that our storage medium would be CD-ROM, because of consistently bad long-term experiences with data storage on magnetic tapes, and because we wanted to learn a bit about CD-ROM writers. We are not concerned with the relatively low data rates of the writers, as we can depend on them to improve, and in any case we can use multiple writers.
- **Security.** Should the CD-ROMs containing network traffic become available for unsupervised inspection, either intentionally or by larceny, it is critical that the data stored on them be protected with strong cryptography. Accordingly, our design goals acknowledge the possibility of loss of

physical control by assuming the worst, anticipating potential publication of the encrypted data. It is also vital that the data be organized in such a way that some subsets of the traffic can be revealed without exposing others. Ideally, we would like to publish keys that unlock certain data on a given CD-ROM, without the possession of those keys exposing other data on it.

We observe that our goals of commodity and completeness are in tension, particularly at network speeds above 10 Mbps. Our goal is to construct a vault that can store all packets on a typically loaded 10 Mbps Ethernet network, and to depend on faster hardware to improve the rate at which packets can be acquired.

3. Architecture

A critical question is whether a single commodity machine can accept packets from the network, encrypt them, and write them to CD-ROM without becoming overloaded. Early experiences with bursty Ethernets coupled with the real-time requirements of CD-ROM recorders† convinced us that two machines would be necessary.

The packet vault hardware is composed of two 133 MHz PCI-bus Pentium machines interconnected via a private 100 Mbps Ethernet. One machine (the "listener") is connected to the network being monitored and is used to capture and encrypt packets, which are then sent over the private Ethernet. The listener never stores packets on magnetic disk.

The other machine (the "writer") receives encrypted packets and assembles them on magnetic disk for subsequent writing to CD-ROM. The two magnetic disks on the writer are attached to a common SCSI bus. A second SCSI bus dedicated to the CD-ROM recorder (CD-R) prevents bus contention. Figure 1 shows the hardware architecture of the packet vault.

We chose UNIX for both listener and writer for its familiarity and flexibility. OpenBSD was chosen for the listener for its kernel packet filtering support; early availability of CD-R drivers dictated the choice of Linux for the writer.

We use BPF [1] on the listener to capture all packets seen on the 10 Mbps network being monitored and write them to an *accumulator* file in a memory file system (MFS [2]). We modified the BPF code to pass

† Our CD-ROM recorder, like all early commodity recorders, possesses a small (512 KB) data buffer and thus requires the attached host to maintain a constant data rate during the entire time the CD-ROM is being written; failure to maintain the required rate ruins the CD-ROM.

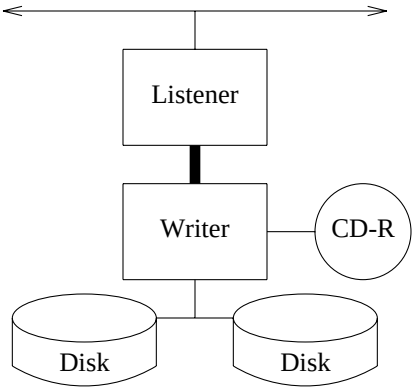

Figure 1

packets directly from the kernel network buffers to MFS, obviating two copies between user and kernel space. A listener process monitors the size of the accumulator file and renames it when it reaches 4 MB in size or after 1 minute has elapsed, which keeps the size of the MFS packet files manageable. The names of the packet files reflect the time of day they were created.

Another process on the listener polls the MFS for new packet files, encrypts their contents, and uses `rcp` to copy the files over the private 100 Mbps link to the writer. Unencrypted data are stored only in the MFS, so in the event of a system failure no unencrypted data remain.‡

When enough packet files have accumulated on the writer to fill a CD-ROM, a background process is spawned on the writer. The writer process generates an ISO-9660-compliant image on magnetic disk containing the packet files and the cryptographic material necessary to permit later recovery of the packet data. The image is written, then purged from magnetic disk. A double-buffering scheme avoids disk contention between image generation writes and subsequent packet file writes on the same physical disk. The packet data path is shown in Figure 2.

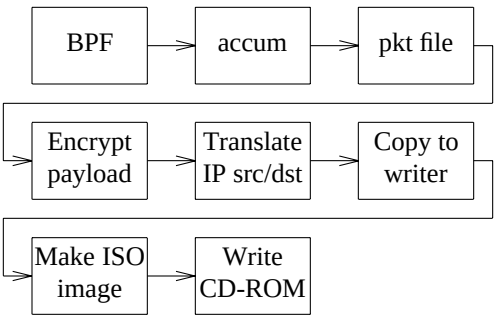

Figure 2

‡ We run the listener with swapping disabled, but acknowledge potential attacks on RAM hardware [3].

4. Cryptographic Organization

The cryptographic organization of the vault follows from our requirement that vault data be publishable — in this way we anticipate the possibility of unrestricted access to a mass storage device filled with vault data. We also endeavor to provide access to individual packet contents with fairly fine granularity.

Our basic strategy is to encrypt all packet payloads; the challenge is to devise a means of associating different keys with different packets at some level of granularity. The ends of the spectrum are unattractive: one key per CD-ROM risks a serious breach if lost, while managing a different key for each packet becomes unmanageable.

Our unit of granularity for associating packets with keys is the *conversation*, defined as a set of packets with the same pair of source and destination IP addresses. Including port numbers would offer finer control, but would also require special treatment for non-TCP streams and create problems with port-agile applications.

Each CD-ROM *volume* holds sufficient information to reconstruct the packet traffic it stores, thus no ancillary information need be managed. We use a multi-level encryption scheme. Symmetric key encryption is used to seal packet payloads and any additional information necessary to reconstruct the packets (explained below). Asymmetric key encryption is used to encrypt the symmetric keys. A trusted third party such as the Regents of the University of Michigan holds the private key. Figure 3 shows the cryptographic organization on CD-ROM.

Our implementation uses 1024 bit PGP [4] for asymmetric key and DESX [5] for symmetric key encryption. Starting with Karn's DES implementation [6] we added both pre- and post-whitening steps for each block:

$$DESX_{k.k_1.k_2}(x)=k_2 \oplus DES_k(k_1 \oplus x)$$

DES encrypts 64-bit blocks, so this requires $64 + 56 + 64 = 184$ bits of key material, and conservatively extends the effective key length of DES in our environment to at least 95 bits with respect to key search (in the sense of Kilian and Rogaway [5]), while adding a trivial amount of computation to each block encryption.†

† If an attacker could obtain all the plaintexts for all encrypted packets on a volume, and if the average packet length is 100 bytes, this would yield 6 million plaintext/ciphertext pairs. Rogaway's effective key length expression becomes $55+64-1-\log_2(6\times10^6) = 95$ bits [7].

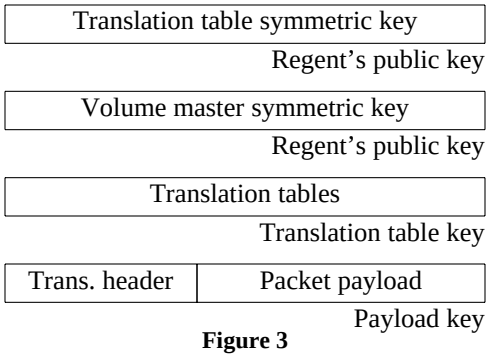

Figure 3

To hinder traffic analysis, we obscure source and destination addresses by substitution. A translation table mapping real to substituted addresses is encrypted with DESX using a *translation table key* K_T unique to each volume. To speed up searches for specific conversations, a second table holds all pairs of translated addresses for which at least one conversation exists on the CD-ROM. The absence of a given pair of addresses in the second table means the CD-ROM contains no packets of that conversation, obviating an exhaustive search to establish this fact. Both translation tables are written to CD-ROM.

A key is constructed for a given conversation by combining the concatenated, untranslated source and destination IP addresses with a 192-bit *volume master key* K_V using exclusive-or, and then using DESX in CBC mode to encrypt a 192-bit constant with the combined value:

$$K_{C_i}=DESX_{K_V \oplus (SA_i \,||\, DA_i)}(CONST)$$

The resulting 192-bit *conversation key* K_{C_i} is used to encrypt packet payloads of the conversation:

$$C_i = DESX_{K_{C_i}}(P_i)$$

A new volume master key and translation table key are generated for each volume. Currently, they are computed from previous keys:

$$K_{V_{i+1}} = DESX_{K_{V_i}}(K_{V_i})$$

$$K_{T_{i+1}} = DESX_{K_{T_i}}(K_{T_i})$$

where K_{V_0} and K_{T_0} were randomly generated. This scheme does not exhibit good long-term randomness; we plan to replace this with a practically strong random data generator [8].

A new pair of PGP keys are generated per vault instance. The public key is used to seal the volume master and translation table keys before they are written to CD-ROM.

Finally, we have built a rudimentary decryption engine that reconstructs all packet traffic stored on a

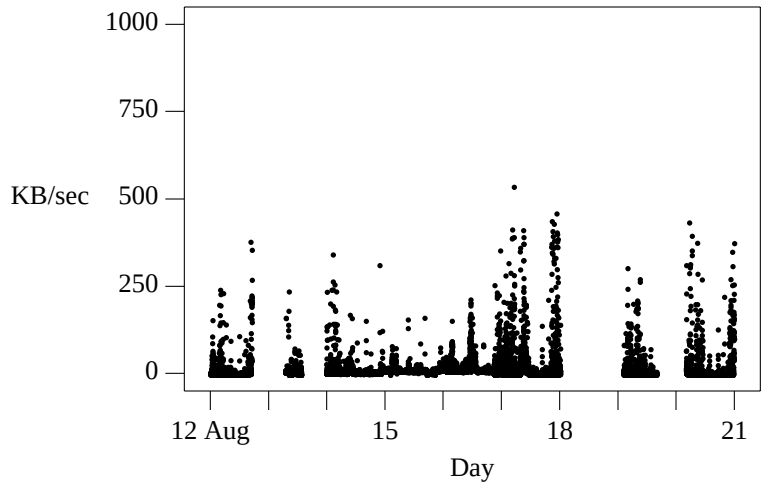

Figure 4 This graph shows vault throughput measured in kilobytes per second during the period 12–21 August 1998. Because values are averaged in 30-second intervals there is some peak clipping; the maximum observed value is 1.2 MB/sec.

CD-ROM given the private PGP key of the vault that created it. We have used the engine to verify the implementation of our cryptographic organization.

5. Experiences

The packet vault has been operational for the last year, irregularly collecting packets from a 10 Mbps Ethernet. The network is usually lightly loaded but there are periods when experimental video work causes traffic to exceed 70%. During the period 12-21 August 1998 we operated the vault continuously, collecting about 7.7 GB on 15 CDs. There were four interruptions of significant duration caused by vault failures during this period. Figure 4 shows a trace of the vault throughput.

The major challenge in the construction and operation of the vault has been systems engineering and integration. Bottlenecks discovered along the way were removed until the vault could handle the incoming network traffic. For example, it was discovered that passing packets in and out of the kernel from BPF to MFS was too slow, so we modified the listener's kernel to skip the kernel/user space copies.

Disk usage on the writer must be monitored closely because of the large data volumes. Currently, the vault does not clean up when interrupted. To achieve reliable operation on restarting, six locations spread across both machines must be checked for abandoned temporary files.

The data path consists of several stages, some of which process data in parallel, others sequentially. Payload encryption and network copying are the most costly operations in this pipeline, yet both of these operations occur sequentially. Generating the image and writing the image to a CD are also costly, but as larger buffers are available for these steps only the average throughput is of importance.

If the sustained input rate exceeds the throughput of any stage in the data path, eventually some buffer becomes exhausted and the vault fails. The first failure is almost always caused by the MFS filling up, which crashes the listening process. Experimentally, with a 70% utilization of the source Ethernet, the vault crashes after about two minutes. Increasing buffer sizes is of limited practical value; doubling the memory allocated to MFS extends this time to four minutes.

At 70% network utilization, while the writer is busy generating a CD image, its disk and processor utilizations increases dramatically, and the `rcp` time increases by a factor of two to three. It takes about 7 minutes to generate an image under these conditions. A bug validated our assumption that double-buffering was needed: a failure to toggle the drive on which the image was being created resulted in packet files for every other volume being written to the same disk on which an image was being built; the resulting overload backed up the data path and crashed the vault.

The other obvious target for performance optimization is the encryption code. We use a machine-specific implementation of the DES code compiled with full optimization and aggressively cache the DES key schedules. These changes speed up the encryption task by over 80%, but opens the door to a denial of service attack by an adversary who manufactures packets that defeat the caching.

6. Discussion

The focus of this work is the creation of a cryptographically secured record of packet activity on a given subnet. The usefulness of such a record is in many ways dictated by the evidentiary requirements of the legal system. Our ability to construct an accurate record must also take into account the creativity and persistence of the adversary, which we consider to be nearly omnipotent. However, we do not address the threats outlined in Schneier and Kelsey [9], in which logging takes place on physically insecure systems; we assume the vault to be under strict physical and administrative control.

6.1. Legal issues

A study conducted by the Office of Policy Development and Education at the University of Michigan identifies a number of thorny legal issues connected with operation of the packet vault [10].

University of Michigan Policy forbids the interception of electronic mail without consent or a court order. Even in the absence of this policy, it is conceivable that a court would find that the vault is intercepting electronic mail under Title I of the Federal Electronic Communications Privacy Act (ECPA). Interception for research purposes might not fall under the so-called "system administrator exemption," which permits interception in the normal course of business or as necessary to protect the rights or property of the service provider, and could therefore be unlawful. In a similar vein, the Family Educational Rights and Privacy Act (FERPA) prohibits disclosing student records to anyone who does not have a specific need to see them.

More generally, the courts have have read the First Amendment of the United States Constitution to prohibit government action that would tend to discourage citizens from speaking their minds. This "chilling effect" applies here, as awareness of the vault's presence would tend to limit free speech by those whose subnets are being monitored.

The vault is a research instrument, so its use is under the purview of the University's regulations on research involving human subjects. While most such research requires informed consent, and the signing of a consent form, it is possible to get an exemption from a University Institutional Review Board. Exemptions can be granted to projects that involve little risk to subjects or where informing the subjects of the nature of the experiment could bias results. Since anyone who sends email to a user on a monitored subnet would arguably be a research subject, obtaining consent from all of them would be problematic.

Other issues potentially raised by use of the vault include increasing the likelihood of copyright violations, bypassing an institution's policies with respect to creating permanent records subject to Freedom of Information Act (FOIA) requests, increasing the likelihood of civil discovery "fishing expeditions" against the material contained in the vault, weakening users' Fourth Amendment protections against search and seizure, and increased exposure of the vault's operators to civil liability.

The question of whether encrypted text is legally the same as clear text has no definitive answer. In some cases, the purported protection offered by encryption is irrelevant; for example, it is likely that copyright infringement takes place when a work is copied, not when it is read.

The study recommends that, at a minimum, all users be notified of the vault's existence. This resolves some of the above legal issues. However, notification would not cure First Amendment "chilling," nor would it address the FOIA or FERPA issues. In addition, it is not clear how to obtain consent from remote correspondents of local users.

A recommended stronger form of consent would allow users to volunteer to be monitored, but requires us to separate on different sets of subnets those users who consent to monitoring and those who do not or to modify the vault to discard certain packets. Both approaches are problematic.

Other recommendations include physically securing all archival materials, maintaining an access audit trail, capturing fewer types of packets, and using the vault only for investigation of specific, ongoing security incidents.

For these reasons, we have chosen not to attach the vault to any production subnets at Michigan, nor to gather many packets. The data we do have were collected on a semi-private CITI subnet for a limited period after all users of the subnet were notified in advance.

In corporate environments, by contrast, the repeated refusal by the Congress to pass laws restricting workplace monitoring suggests that a business is free to monitor workers' communications on its computer systems without consent or knowledge. In fact, in securities trading environments, Wall Street regulations require such monitoring. Use of our vault is less controversial in these environments (at least for now).

6.2. Limits of DES

Recently, the Electronic Frontier Foundation's DES cracker and a worldwide network of personal computers jointly won RSA Data Security's DES Challenge III, obtaining the encryption key to a DES-encrypted message in 22 hours via a brute-force search of the key space [11]. Two previous challenges were successfully cracked by similar methods. Since the vault uses DES at the core of its encryption strategy, these events call the security of the data stored on vault CD-ROMs into question, especially in the long term.

First, we believe our use of DESX inhibits the use of brute-force DES crackers because it is difficult for an attacker to derive a plaintext/DES-ciphertext pair from a set of plaintext/DESX-ciphertext pairs obtained by a chosen-plaintext attack [7]. However, we have not quantified the effect of DES's rapidly declining strength on our cryptographic organization.

Second, we can replace DES with more secure triple-DES; this increases the key length of all symmetric keys as recommended by Blaze *et al* [12] and is a straightforward modification. However, triple-DES is roughly three times as slow as DES; even though processors today are more than three times faster than they were when we started our project, the additional encryption cost may be prohibitive.

Finally, we can look to other recent proposed encryption algorithms and the results of the Advanced Encryption Standard effort to deliver a more secure encryption algorithm for our vault, although this is necessarily a long-term prospect.

In any event, recent developments have shown that ciphers once considered secure are rapidly being broken as technologies and analysis techniques mature. It is not reasonable to assume that any cipher will be strong enough to withstand decades of determined attacks, which implies that loss of physical control of the vault media will lead to exposure of the data they carry.

6.3. Limits of Passive Protocol Analysis

Ptacek and Newsham point out a shortcoming in passive protocol analysis due to the inability of an intrusion detection system to determine accurately what is happening on networked machines [13]. They identify three classes of attacks: insertion, in which the detector is made to see traffic that the victim does not; evasion, in which the victim sees traffic the detector does not; and denial of service, in which the detector is fed traffic designed to cause it to fail.

The packet vault is largely immune from these attacks — because the vault obtains packets directly from the link level device driver, it does nothing beyond reading and storing each packet that arrives on the interface. Fragment reassembly, management of TCP connection state, *etc*. are left to the analysis phase after the CD-ROMs are written. This causes attacks on the vault by, say, deliberately overlapping fragments to fail, as the vault does not reassemble them; further, the complete evidence is stored for later analysis. There is some potential for denial-of-service attacks, including the one mentioned earlier that defeats the caching of key schedules.

As long as the recording rate exceeds the arrival rate, then the packet vault defeats evasion and denial of service attacks. Insertion attacks are possible, but permanent storage of all packets permits later replay on appropriately instrumented test gear.

6.4. Evidence handling

Sommer outlines general principles for the production of reliable, computer-derived evidence [14]:

- the scene of the crime must be "frozen"
- there must be continuity of evidence
- all procedures used in examination should be auditable

The packet vault records onto CD-ROM — an immutable material that effectively "freezes" the evidence — all data that traverse a snoopable subnet. While it is possible that some packets traversing the network during periods of peak load are not seen by the vault, its architecture precludes the generation of spurious packets, *i.e.*, the vault does not manufacture evidence. The vault thus provides evidence that can be used to support other materials, such as audit logs.

Continuity of evidence is indicated by the data handling architecture of the vault. The monotonically increasing time-stamped sequence of stored packets lends further support for continuity of evidence. Including a digital signature with the CD-ROM contents would help prove the authenticity of any CD-ROM that purports to have been generated by the vault.

The vault source code and, potentially, the contents of the CD-ROMs are available for public inspection, which allows the procedures to be audited.

6.5. Future work

The next major step involves focusing on intrusion detection methods, replaying vault data in a virtual network testbed. Better administrative and fault-handling scripts are also needed for graceful shutdown and restart of the vault. An occasional inability of the writer to allocate buffer space for the private Ethernet link

remains to be resolved. The high disk loads caused by creating an ISO-9660 image *en masse* could be ameliorated by constructing the image incrementally. We plan to replace our hastily constructed key generator with a practically strong random data accumulator and generator [8]. We will investigate the issues in replacing DES by triple-DES or another cipher to provide a more secure cryptographic organization for our vault. Determining and recording the number of packets dropped during the generation of and digitally signing each CD-ROM would improve the evidence handling capabilities of the vault. Finally, these and other steps are necessary to convert the vault from a research instrument to a highly-available packet capture and storage mechanism.

7. Acknowledgements

We thank Mike Stolarchuk for his contributions to the architecture of the packet vault. He also wrote the BPF layer modifications, and provided invaluable systems engineering assistance. Dan Boneh suggested the conversation key mechanism. Joe Saul thoroughly investigated the legal issues in operating the vault. We thank Dan Geer for his helpful review and commentary. This work was partially supported by Bellcore.

8. References

1. Steven McCanne and Van Jacobson, ‘‘The BSD Packet Filter: A New Architecture for User-level Packet Capture,’’ pp. 259–269 in *Proc. of Winter USENIX Conf.*, San Diego (January, 1993).
2. Marshall Kirk McKusick, Michael J. Karels, and Keith Bostic, ‘‘A Pageable Memory Based Filesystem,’’ pp. 137–143 in *Proc. Summer USENIX Conf.*, Anaheim (June, 1990).
3. Peter Gutmann, ‘‘Secure Deletion of Data from Magnetic and Solid-State Memory,’’ pp. 77–89 in *Proc. of Sixth USENIX Security Symp.*, San Jose (July, 1996).
4. William Stallings, ‘‘Protect Your Privacy: The PGP User’s Guide,’’ *Prentice-Hall*, New Jersey (1995).
5. Joe Kilian and Phillip Rogaway, ‘‘How to Protect DES Against Exhaustive Key Search,’’ pp. 252–267 in *Advances in Cryptology - Crypto ’96, Lecture Notes in Computer Science*, ed. N. Koblitz, Springer-Verlag (1996).
6. Phil Karn, `karn@unix.ka9q.ampr.org` (December, 1995).
7. Phillip Rogaway, *RSA Laboratories’ CryptoBytes* **2**(2) (Summer, 1996).
8. Peter Gutmann, ‘‘Software Generation of Cryptographically Strong Random Numbers,’’ pp. 243–257 in *Proc. of Seventh USENIX Security Symp.*, San Antonio (January, 1998).
9. B. Schneier and J. Kelsey, ‘‘Cryptographic Support for Secure Logs on Untrusted Machines,’’ pp. 53–62 in *Proc. of Seventh USENIX Security Symp.*, San Antonio (January, 1998).
10. Joseph M. Saul, Peter Honeyman, and Virginia Rezmierski, ‘‘Policy Issues Related to Network Monitoring: The Secure Packet Vault,’’ Unpublished, Ann Arbor (July, 1997).
11. Electronic Frontier Foundation, in `www.eff.org/DesCracker/`.
12. Matt Blaze, Whitfield Diffie, Ronald L. Rivest, Bruce Schneier, Tsutomu Shimomura, Eric Thompson, and Michael Wiener, ‘‘Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security ,’’ in `www.counterpane.com/keylength.html` (January, 1996).
13. Thomas H. Ptacek and Timothy N. Newsham, *Insertion, Deletion, and Denial of Service: Eluding Network Intrusion Detection*, Secure Networks, Inc. (January, 1998).
14. Peter Sommer, ‘‘Computer Forensics: an introduction,’’ in `www.virtualcity.co.uk/vcaforens.htm` (1997).