DRAFT

Recent events have re-emphasized the need for quick destruction of sensitive material. The use of computers has been suggested in order to replace paper files with computer files thought to be more easily destroyed.

This paper examines the premise of easily destroyed computer files, and illustrates some practical limitations. Two scenarios are examined: the use of an on-site minicomputer; and, interactive use of a remote computer. More importantly, the hypotheses underlying the need for quick destruction are examined from a decision theoretic viewpoint. This provides an interesting viewpoint and argues for cryptographic protection of computer files as the most effective way of providing quick "destruction".

Unarguably, there exists a need for the safe and rapid destruction of sensitive material under emergency conditions. Recent events have prompted a re-examination of the use of computer technology to provide that emergency destruction capability. The hypothesis is that reliance on computer management of information will: effectively eliminate paper files; and, the computer files that replace these will either: be located remotely in safe haven; or, be easily destroyed. These assumptions are examined in detail, last first.

Easy destruction of computer files springs readily to the mind of anyone who has had to rely on computer storage of data. A confluence of factors make the accidental "destruction" of our data a too-frequent occurence. Whether these accidents are caused by hardware "glitches", software "bugs", or operator error, the incidence of unusable data is quite real. It is worthwhile to look in more detail at such events, and at the normal procedures which protect against such accidental destruction, thus minimizing the impact on everyday work. When the computer informs us, arcanely: "Abend / parity error / file not found" or the like, it is saying only that the data is not normally recoverable (i.e., it may not be recoverable in its entirety, and extraordinary techniques might be needed for its retrieval.) Such examples of quick "destruction" provide little comfort in the face of a determined, and modestly sophiscticated foe with reasonable time to recover the data.

More importantly, however, the frequency of such accidental unavailability of data leads to operating procedures which keep "backup" copies of the files on "backup" disks, with "backup" tapes ... and, typically, with hard-copy, paper "backup". We should not lose sight of the fact that the computer is the largest generator of paper in any organization.

Suppose, however, we have conquered the accidental unavailability problems, (an important design constraint for such an application) thus obviating the need for extensive paper backup. Now we can concentrate on purposeful destruction of computer stored data. Three major types of destruction are considered: electrically altering the appearance of the data; electrically/magnetically destroying the data; and physical destruction of the storage media.

> electrical alteration refers to the practice of overwriting the data with new, meaningless data which obscures the original. This technique can be used with (nearly) all types of computer storage (solid state memory, core, disk, tape) and requires the storage to be on-line (another important design constraint).
>
> electrical destruction depends on the fact that some computer storage retains data only when powered (solid state memory) so that powering down the memory destroys the data.

25X1A

[redacted]

> physical destruction is the mechanical/chemical disintegration of the media upon which the data are stored, and works best with smaller, less dense storage media, the less rigid (floppy disk as opposed to hard Winchester technology) the better. The storage media must be demounted.

Procedurally, each form of destruction would proceed in turn for ultimate assurance against determined recovery: electrically alter, demount and degauss, and then disintegrate.

The reconstitution would, of course, depend upon the stage to which destruction had been carried. The media might have to be replaced, perhaps from stock. Replacement of information, however, puts a severe demand upon the telecommunications, precisely at the time when other demands are being placed upon it, for command and control, and reporting, and precisely at the time when it is likely to be severely degraded, due to environment or host government actions. Ponder the notion of writing a disk memory over a 100wpm teletype circuit share with other traffic, and with message overhead and error control which effectively halves the bandwidth. At an effective

25bps, a modern 300mb disk would require over 3 years to re-create.

The preceding analysis has been largely directed to on-site computer use. Operational considerations attendant on use of remote ADP, traditionally dictate reserve paper files as well as redundant, high-bandwidth (expensive) telecommunications. Otherwise, inevitable communications outages seriously degrade day to day operations. Remember, too, that good telecommunications are least likely to be available precisely during the periods in which it is necessary to have a quick destruction window, and there will be an irresistible temptation to keep backup paper files. Thus, the quick destruction goal must be achieved in harmony with adequate normal operation, and acceptable, if degraded emergency operation. This must mean quick reconstitution, or guaranteed all-weather remote access to safe haven computing, which again implies expensive telecommunications.

As we have seen, destruction and reconstitution must be considered together. Let us formalize this slightly. There ultimately will be a decision between two alternatives: DESTROY, or RETAIN. Applicable to that decision are two states of the world: Destruction Really Required, or Destruction Not Really Required. In the face of imperfect knowledge, and imperfect decision makers, four outcomes are possible:

a. We destroyed when it was required;

b. We retained and destruction was not really required;

c. We retained when destruction was required;

d. We destroyed when destruction was not really required.

Costs/benefits can be assigned to each outcome. The first two outcomes, a and b, were good; the last two outcomes, c and d, were

unfortunate. This is shown below.

a decision is made to:

| when, in fact, | DESTROY | RETAIN |
|---|---|---|
| Destruction REQUIRED probability = P | (a) GOOD OUTCOME Loss prevention . . | (c) BAD OUTCOME Possible loss of assets & other grave damage |
| Destruction NOT REQUIRED probability = (1-P) | (d) BAD OUTCOME Cost a function of time/expense to reconstitute | (b) GOOD OUTCOME Normal operations unimpaired |

Current events focus our attention on the cost of outcome c. However, it is just as important to focus on the cost of outcome d because our decision as to whether to destroy or retain is influenced by the entire outcome cost/benefit matrix as well as our apriori assessment of the probability that destruction will really be required.

> Uncertainty in assessing the probability that destruction will, in fact, be required, coupled with the cost of outcome d (the cost of reconstitution) tempts us to postpone as long as possible the actual decision. And, the desire to postpone "as long as possible" the decision to destroy, forces us into demanding, say, a reduction in holdings to a one-hour destruct cycle.

Thus, reducing the reconstition time is a very important, sometimes overlooked design constraint ... not only because it impairs subsequent operations, but because it impacts our decision as to when to destroy.

Encrypted storage of information allows us to add a new type of "destruction" to our list above:

> Key destruction - destroying the crypto keys which had been used to encrypted a data file renders that data unintelligible; the key volume, being much smaller than the volume of total information allows quicker destruction by

any of the previously discussed techniques. More importantly the small volume of key material can be quickly reconstituted, thereby quickly reconstituting the information.

Encrypted storage of information, no matter the media on which it is stored (disk, tape, cassete, etc.), also affords much better protection overall, inasmuch as it is the smaller amount of key which must be most strongly protected.

In assessing the utility of encrypted storage, several constraints must be examined. First, there is the question of how secure the encrypted material is? How resistant is it to concerted attack? If not resistant to attack, we would certainly not wish to hand to a determined adversary a disk-full of data. For the field station, however, it is useful to remember how the bulk of the data was received, or sent ... by encrypted telecommunications. Any realistic threat assessment must assume that the determined adversaries already have disks full of our encrypted data. We are, then, no more constrained by the encryption algorithm's robustness (or lack thereof) for static storage than we were for communication of that same information.

A second constraint is the increased key management which would be required. Today, the domain of keys managed (generated, indexed and stored for retrieval, and destroyed) is limited. Use of cryptography for static storage would, perhaps: double the amount of key which would need to be generated (and, perhaps halve the design life of an algorithm); double the amount of key which would need to be destroyed, in steady state; and increase more markedly the amount and velocity of index, storage, and retrieval.

Another constraint which would need to be examined is the possible difference between keys for static storage, and transient communication use. (Although the image of our adversary's National Storage Agency full of statically stored, encrypted data should provide the Occam's razor for this perceived dilemma.)

A third constraint is the granularity with which keys are assigned to data. Some alternative assignment strategies include:

- key per physical storage space ... key per disk
- key per logical storage device
- key per physical/logical record
- key per entity/attribute in a database

We move from the former to the latter strategies as our goal changes from simply overall protection and quick destruction/reconstitution to compartmentation and command privacy at an elemental level.

Another design factor is the use of a master key to encrypt the

individual keys required by our choice of granularity, which enforces the command privacy and appears to reduce still further the amount of material which needs immediate destruction. Whether this weakens the overall security afforded is equivalent to our faith in deterministic key generation by (presumably) known algorithm from a starting seed. (A blade for Occam's razor.) The desirability of repeatedly changing the master key is a function of the depth of the algorithm.

Another constraint to be investigated is the problem of updating data files and retrieving a portion of a file if a key of coarse granularity has been decided upon. Choice of design strategy here is the same as the dilemma of having a sequential-access device of finite speed serve as a random access memory, and the same blocking issues suffice. We need be no more constrained in using a encrypted disk than a clear text disk, inasmuch as the disk is, at best, block-sequential. (Another blade for Occam.) Another proven strategy for dealing with the update problem is that used with write-once media (from paper tape to optical disks to journals) or extremely large, sequential files... update by "posting" transactions, and (infrequent) batch reorganizations.

A first cut design, then, argues for a policy-high, red computer with all dynamic, solid-state memory; all other memory devices are I/O devices with crypto on the red side of error control protocols (e.g., CRC calculation.) (One more for Occam ... think of the writing-to reading-from disk as end-to-end encryption with a variable delay communications line.)