2

The Puzzling Science of Information Integrity

G. J. Simmons
P.O. 365
MN 87047 Sandia Park, USA, Phone/FAX No:(505) 281-3590

Abstract

The science of information integrity is concerned with preventing deception and/or cheating in information dependent systems; or failing that, to at least detect deception and assign responsibility if it does occur-- where the means of deception is solely the manipulation of information. In other words, information integrity is supposed to make it possible to trust the correct functioning of the system, even though (some of) the inputs may be untrustworthy. Typical deceptions might be; gaining unauthorized access to files or facilities, impersonating another user or forging his digital signature, disavowing a message that was actually sent (and received) or else falsely attributing a message to a transmitter who did not originate it, etc. Solutions to problems of this sort, while important in the classical two party communications setting, are crucial in a multiparty network setting where the number of participants may be essentially unlimited, as are the types and objectives for deception

Systems and protocols devised to protect against deception are fundamentally different from all others. For example, the specifications for a piece of communications gear might specify the natural environment in which the equipment is supposed to operate, such as voltage and temperature extremes , the shock, vibration and noise environments it must tolerate etc. The equipment can then be tested to verify that it meets these specifications and certified that it does. An information integrity protocol, however, can never be certified in this same way, since the hostile environment is not nature but rather an intelligent opponent(s) who can be expected to exploit his knowledge of the system and all information he may acquire about it and about the actions of the other participants to maximize his chances of success in cheating the system. He may act in ways not anticipated by the designer, or join forces with other participants to form cabals not planned for in the design. Nature, while it may present a hostile environment, is unknowing and impartial. The human opponent is knowledgeable and capable of finding and exploiting any weaknesses the system may have.

In all disciplines a difficult and as yet unsolved problem can be properly described as puzzling. However, in most disciplines, once a solution is found the puzzle is solved. Information integrity,

R. Posch (ed.), *Communications and Multimedia Security*

to initiate the controlled action without the other shareholders concurrence? and if they can, can they also produce a bona fide arbiter's certificate fraudulently indicated that he had concurred This is by no means an exhaustive list of the conceivable types of deception in this simple protocol, but it should give the reader some feeling of what is involved when the trustworthiness of everyone involved must be considered as suspect. In a real world application, all possible deceits need to be recognized and considered, but fortunately (for the sanity of the designer) many can be dismissed as either being too unlikely of occurrence to be of concern, or else simply ruled out as deceits that a particular protocol can't deal with.

The two man control protocol also provides an example to illustrate how the information integrity primitives we will introduce later can be combined to construct protocols. Two man control is a simple example -- indeed the simplest possible example -- of a shared secret scheme. In spite of this apparent simplicity, things may not be so simple. If an unconditionally trusted authority exists to generate the two shares and to distribute them in secrecy to the two shareholders, and if there is no need for the shareholders to be able to prove to themselves or to any one else that they have been given bona fide shares, then the solution is indeed simple. If, however, the shareholders demand that they be protected from either the issuing authority exposing their shares, or of him misusing them to initiate the controlled action and then blaming them, the problem becomes very difficult. Ingemarsson and Simmons devised a protocol with which parties who mistrust each other can set up a shared secret scheme that they must logically trust without the aid of a mutually trusted issuer. The problem here is one level more difficult than the one that was solved by Ingemarsson and Simmons in that a third party who has the authority to delegate the distributed capability to initiate the controlled action must also be involved. In the Ingemarsson and Simmons scheme the shareholders end up being certain that they each hold a bona fide share of a secret which they do not know and which none of the other shareholders know either, but which they know was jointly determined by them. In the present example, the shareholders want to be certain that the shares are completely indeterminate to the issuing authority, even though they are not free to determine (jointly or in combination) the secret itself. Simmons devised a key distribution protocol, the main feature of which was that two parties, say A and B, would interact to determine a "random" number whose value was totally indeterminate to each of them in advance and which only A would know when the protocol was completed. B, however, even though he didn't know the number they had jointly generated, could verify that A was using it as the key in a cryptographic protocol. The present problem is similar, but more complex, in that each of the shareholders must end up in possession of a random number (share) which only he knows, but which must be related to the secret being shared, and hence to the other parties share, and whose (joint) use can later be verified by others.

The shared secret scheme is itself often more complicated than a simple two out of two concurrence -- typically requiring that any pair of shareholders, out of several, be able to initiate the action: a k out of m threshold scheme where $k < m$. Clearly each shareholder must keep his share secret. If the information content of a share is small enough for them to be able to recall it from memory, then memorization may suffice. However, this limits security to one in a trillion or so. If higher security is needed, then one must find a cryptographic technique amenable to mnemonic key storage and of adequate security. If it is necessary for the shareholders to be able to prove to themselves and to others that they hold bona fide shares, without eroding the security of the controlled action, then a difficult extension of the notion of zero knowledge proof to zero knowledge distributed proofs is required. A simple example of what we are talking about here

would be to devise a protocol with which A and B can be given "shares" with which they can prove to others that they jointly possess all of the information needed to factor a very large composite integer, although neither of them alone has any improved chance of factoring the integer than does an outsider who only knows the large integer in question. The reader should recognize that two types of shared capability are being discussed; the shareholders share a secret (piece of information), but they also share a function, in this case the functional ability to prove that they could do something which they haven't done. The two man rule can also require other primitives in the construction of a protocol. The arbitration function, whose purpose is to assign responsibility (for initiating the controlled action), is dependent on a distributed signature being produced by the shareholders when they exercise the capability they have been given. There are also aspects of authentication, notarization, time stamping etc. The point of this discussion was to illustrate both how much is needed in the way of primitive building blocks to construct information integrity protocols, and to suggest what some of these primitives might be.

Having set the stage for a discussion of the science of information integrity, all that is possible in this abridged introduction will be to sketch the essentials of the three points mentioned at the beginning. Obviously, the most important thing to clearly understand is what the functions are that information integrity protocols are designed to achieve. As we said earlier, every such function has as a mirror image at least one deception it is intended to thwart. The following table summarizes some of the principle information integrity functions. We tabulate the functions rather than the deceptions, because the one can be described in telegraphic style, while the other cannot.

2 A PARTIAL LIST OF INFORMATION INTEGRITY FUNCTIONS

1. Identification
2. Authorization
3. License and/or certification
4. Signature
5. Witnessing or notarization
6. Concurrence
7. Liability (acceptance or establishing)
8. Receipts
9. Certification of origination and/or receipt
10. Endorsement
11. Access or egress control
12. Validation
13. Time stamp
14. Authentication
15. Voting
16. Ownership
17. Registration
18 Approval or disapproval
19. Privacy (secrecy)
20. Anonymity

Consider certification of receipt for example. A sender wants to be able to prove (in a court of law or to an impartial arbiter) that an important piece of information that he originated or sent to another individual was actually seen or received by that individual. Certified mail which is used for this purpose in much of the Western world does nor really serve this function. The receipt for certified mail merely establishes that something was received and signed for by the addressee, not what was received. An information integrity protocol for this function should do both. It may well be required to do even more, i.e. to certify that a hidden version of a message (cipher) was received and signed for in a way that could be verified by third parties who at the time they certify the validity of the receipt do not know the content of the message involved.

A related problem occurs in the notarization of information. A notary is testifying to the identity and presence of the signatories to a document, as well as to the time and date they appeared to sign the document and to have it notarized; not to the content of the document being notarized. In many digital notary schemes it is easy to create a notary seal to a document that the notary did not see and did not notarize, and indeed which he might not have notarized had he known its content, by getting him to notarize other pieces of innocuous information. Moore devised a whole series of protocol failures of this type for notary schemes.

Chaum has spent much of his career on questions of anonymity which is in a sense the logical dual to several of the functions listed here; identification, digital signatures, certification of origination etc. The Crime Stoppers Hotline concept in the U.S. where tipsters can anonymously report criminals illustrates some of the difficulties of providing anonymity while at the same time assigning liability, in this case delivering a reward for useful information without knowing or compromising the identity of the tipster. In commercial transactions there is frequently the need to maintain anonymity, while at the same time there are all sorts of mischief that can safely be worked from the cover this provides. Some of the more reprehensible information on the internet is a case in point.

The list of possible deceptions is enormous, but the reader should see that each requires considerable discussion -- not amenable to succinct tabulation -- hence the decision to list function instead.

Since the integrity of all information integrity protocols is measured in terms of just how difficult the puzzle is that a would be deceiver must solve in order to work a successful deceit, it is necessary to explain the terminology used to quantify these. A scheme is said to be unconditionally secure, computationally secure, provably secure or of course, insecure, by the following criteria.

1. A scheme is said to be unconditionally secure, if the probability of a would be cheater being successful is independent of the computing power or expenditure he employs.

2. A scheme is said to be computationally secure if upsetting its intended function requires a would be cheater to carry out some computation that in principle is possible, but in which all known methods of execution require an infeasible amount of computation.

3. A scheme is said to be provably secure if it can be shown that upsetting its intended function for any "significant" number of cases implies that some other hard problem -- such as factoring suitably chosen large composite numbers or extracting discrete logarithms in similarly chosen fields

etc -- could be solved with comparable effort.

While both computationally secure and provably secure protocols (or algorithms) depend on problems of known hardness (computational difficulty) for their security, in the first case it is only known that the protocol is secure if the problem is hard, while in the second it is proven that the protocol is secure if and only if the problem is hard. Vernam encryption, perfect shared secret schemes and the authentication codes of Simmons are all unconditionally secure. Williams version of the RSA crypto scheme is provably secure, i.e. for a suitable choice of the primes p and q and with the public exponent 3 he proved that decryption (of any significant number) of ciphers was equivalent to factoring the modulus n = pq. Maurer has devised a number of provably secure protocols and Brickell and McCurley have devised at least two doubly proven secure examples in which the protocol is as secure as the harder of two problems. Most protocols are computationally secure so it is unnecessary to cite examples.

We have now seen what some of the functions are that information integrity protocols are supposed to realize, and how the security of these schemes is measured. As promised we conclude by listing some of the basic building blocks used to construct the protocols. The reader will be left to his own imagination to either parse schemes he is familiar with into these components, or else to assemble components to realize functions for himself. In the complete paper the author will do both of these for a variety of functions.

3 INFORMATION INTEGRITY PRIMITIVES

Secrecy
- Overt (cryptography) / Covert (subliminal)
  - single key
  - two key (aka public key)
  - multiple key

Authentication*
- specified verifiers / public verification
  - without arbitration
  - with arbitration
    - active (arbiter participates initially)
    - passive (arbiter is involved only if arbitration is needed)

Shared Capability (of)
- recovering secret information
- exercising a function not publicly possible

Data Compression or Message Digests
- invertible, i.e. lossless -- ZLW, source coding etc.
- non-invertible
  - mappable -- check sums, CRC etc.

- one way, collision free -- hashing

Identification (of)
- data
- devices
  - intrinsic -- fingerprint
  - attached -- brand
- individuals
  - intrinsic -- fingerprint, voiceprint, retinal scan etc.
  - attributed -- knowledge

Commitment to a ;
- number -- bit etc.
- function

Digital Signatures*
- specified verifiers / public verification
  - without arbitration
  - with arbitration
    - active (arbiter participates initially)
    - passive (arbiter is involved only if arbitration is needed)

Error Detecting and/or Correcting

One-way Functions
- non-invertible
- invertible
  - computationally infeasible
  - trapdoor

Proof
- membership / knowledge
  - zero knowledge
  - limited, but tolerable, exposure (erosion of integrity)
  - full exposure -- to a "limited" audience

Random Generation (of)
- number
- function

---

*Digital Signatures and Authentication are similar, but not identical, functions. The function of authentication is to establish that information is as it is purported to be, i.e. that it has not been modified, substituted, forged etc. subsequent to its generation, while digital signatures are designed to establish the originator (or origin) of the information.

Timing and/or Time Stamping (to establish)
- simultaneity
- sequential
- absolute

Trust Mechanisms (for)
- transfer of trust
- realizing distributed trust

4 BIOGRAPHY

Gustavus (Gus) J. Simmons was born at Ansted, WV, on October 27, 1930. In his senior year of high school he was one of 40 national winners in the Westinghouse Science Talent Search. He attended Deep Springs College, Deep Springs, CA, 1947-48, following which he spent almost five years in the U. S. Air Force as a radar mainteance specialist. He received a B.S. in mathematics in 1955 from Highlands University, Las Vegas, NM and an M.S. in physics (with a minor in mathematics) in 1958 from the University of Oklahoma, Norman, OK. He was the first recipient of ta doctorial studies fellowship (in 1968) from the Sandia National Laboratories, under which he received his Ph.D. in mathematics from the University of New Mexico, Albuquerque, NM, in 1969.