A network security threat in general (even when cryptography is used)

Mike Burmester[a] Yvo Desmedt[b] Gregory Kabatianski[c]

[a]Information Security Group, Department of Mathematics, Royal Holloway, University of London, Egham, Surrey TW20 OEX, U.K., m.burmester@rhbnc.ac.uk

[b]Center for Cryptography, Computer and Network Security, Dept. EE and CS, Univ. of Wisconsin – Milwaukee, P.O. Box 784, WI 53201 Milwaukee, desmedt@cs.uwm.edu

[c]Inst. for Problems of Information Transmission, Russian Academy of Sciences, Ermolovoy 19, Moscow, GSP-4, Russia, kaba@ippi.ac.msk.su

Abstract

The emphasis on cryptography in modern books (*e.g.* [3]) seems to suggest that (if combined with network reliability and anti-jamming techniques) it can guarantee network security. We dispute this by presenting an attack against any network which has been protected using conventional cryptographic techniques and in which there are nodes that do not know the network graph.

For the sake of simplicity we assume that nodes in a network share a secret key with their neighbors and that a conventional cryptosystem is used to protect authenticity of messages. If a node n_i is a neighbor of a node n_j, we assume from now on that *all* traffic between n_i and n_j will be authenticated using the common secret key. If a node n_i is not a neighbor of a node n_j then the nodes n_i and n_j must rely on intermediate nodes to exchange authenticated messages (we do not worry about privacy in this paper). However, if insiders may be dishonest then the receiver, n_j, cannot trust that the message will indeed be authentic even though it is claimed to be. If we assume that at maximum u nodes will be dishonest the theory of network reliability tells us that if the network is $2u+1$ vertex connected[1], the sender will always succeed in having at least $u+1$ identical messages (with the same sequence number) delivered to the receiver through *disjoint paths*. This seems to indicate that the receiver can use a majority vote to decide that the message is authentic, or not. We will see that in an adversarial scenario this result from network reliability does not necessarily carry over.

First of all we observe that nodes should not be allowed to use pseudonyms since otherwise they might be able to create paths which seem disjoint when in fact these are not. Secondly, it is also easy to see that messages must contain as information the path that these have supposedly followed. Indeed, otherwise the receiver is not able to verify that these messages come from disjoint paths. If the sending node knows the network graph, then it can easily compute [1] disjoint paths and replicate sending the message via these paths appending each time to the message the path it

[1]A variant of vertex k-connectivity in which between any two nodes there is either an edge or there are k disjoint paths, is sufficient in our scenario.

needs to follow. However, if the receiver does *not*[2] know the network graph, but the u adversaries do, then they can inject fake messages and specify paths these have to follow. We now elaborate on this.

To be more specific, we assume that all honest nodes n_{i_j} specified in a path $n_{i_1}, n_{i_2}, \ldots, n_{i_l}$ will verify that a received message indeed came from or via the node $n_{i_{j-1}}$, by checking the authenticity. If this is not the case, the node n_{i_j} will not forward the message, nor accept it. We now discuss bogus paths, *i.e.*, originating from a dishonest node, let say $n_{i_{j-1}}$, who claims it originated from n_{i_1}, via n_{i_2}, etc. A message traveling via a bogus path will be forwarded by honest nodes (including n_{i_j}) as long as in the specified bogus path $n_{i_{j-1}}$ is followed by n_{i_j}, where $(n_{i_{j-1}}, n_{i_j})$ is an edge and the dishonest node $n_{i_{j-1}}$ sends the message (with the bogus path) to n_{i_j} and authenticates it with the key they share. The actual message (without the path specification) may be identical to an authentic one being sent (but sent via a bogus paths), or may be different. If the message originated from the sender, *i.e.* is authentic, it will be received by at least $u + 1$ disjoint paths (possibly) among other bogus paths. Otherwise there is a subset of at maximum u nodes such that each path must have gone through at least one of these nodes. It seems hard for the receiver to conclude whether the message is authentic or not. Indeed, detecting authentic messages corresponds to the **NP**-complete Set Packing problem and detecting non-authentic messages to the **NP**-complete Hitting Set problem [2]. However, since we know that the paths will be restricted to one or the other form (the number of bad users is guranteed at most u), it is not obvious to conclude whether our problem is **NP**-complete. The best algorithm we know to decide authenticity uses exhaustive search. So, it seems that when u is say larger than 50 (which is realistic in the internet environment, or when trying to protect against information warfare), exhaustive search takes too long.

It should be observed that the attack does not extend to digital signatures or to certificates. Finally we note that we have no solution to solve this problem, and definitely none when the sender also does not know the network graph, as is often the case in real networks. We question whether a secure network can be obtained when the network graph is unknown and when several insiders could be dishonest without the use of digital signatures.

Acknowledgement We thank Mike Reiter and Stuart Stubblebine for the discussions about certificates.

References

[1] S. Even. *Graph algorithms.* Computer science press, Rockville, Maryland, 1979.

[2] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, San Francisco, 1979.

[3] C. Kaufman, R. Perlman, and M. Speciner. *Network Security.* Prentice-Hall, Englewood Cliffs, New Jersey, 1995.

[2] If the receiving node knows the network graph, it can use the same deterministic algorithm [1] to find the exact same disjoint paths as the sender has found and eliminate all incoming messages (with the same sequence number) that have arrived through other paths.