Byzantine Modification Detection in Multicast Networks using Randomized Network Coding

Tracey Ho*, Ben Leong*, Ralf Koetter†, Muriel Médard*, Michelle Effros‡ and David R. Karger*[1]
*Massachusetts Institute of Technology, †Univ. of Illinois, Urbana-Champaign, ‡California Institute of Technology
e-mail: *{trace@ , benleong@, medard@, karger@csail.}mit.edu, †koetter@uiuc.edu, ‡effros@caltech.edu

Abstract — **We show how distributed randomized network coding, a robust approach to multicasting in distributed network settings, can be extended to provide Byzantine modification detection without the use of cryptographic functions.**

Distributed randomized network coding is a flexible, robust approach for multi-source multicast in distributed network settings. In this technique, nodes independently select random linear mappings from inputs onto outputs over some finite field. This achieves any feasible connections with probability tending to 1 as the field size grows. The aggregate linear combinations can be communicated to receiver nodes as coefficient vectors which undergo the same operations as the information signals. Such information allows receiver nodes to decode the source messages if they receive enough independent linear combinations [1]. This approach achieves efficient shared use of multiple paths, giving greater robustness to link failures and random coding errors as excess capacity in the network increases [2]. Reference [3] describes a practical packet-based implementation which divides source packets into generations within which linear combinations may occur.

In this paper, we show how this approach can be extended to detect Byzantine (i.e. arbitrary) modification of data by malicious or compromised nodes. This is particularly useful in overlay or ad-hoc multicast settings where end hosts forward information to others. Other Byzantine fault detection approaches have included message authentication codes [4] and signed digests [5]. We consider a packet-based randomized network coding scheme, where some hash symbols, calculated as simple polynomial functions of the source data, are included in each source packet. Receiver nodes check if decoded packets are *consistent*, i.e. have matching data and hash values. Additional computation is minimal as no cryptographic functions are involved. Detection probability can be traded off against communication overhead, field size (complexity) of the network code and the time taken to detect an attack.

The only requirement is that receiver nodes obtain one or more unmodified packets whose contents were unknown to the Byzantine attacker at the time of design of the modified packets; we will refer to such packets as *good*. This expectation is reasonable given the distributed randomness and path diversity of network coding. Depending on the application, various responses may be employed upon detection of a Byzantine fault, such as collecting more packets from different nodes to obtain a consistent decoding set, or employing other more complex Byzantine agreement algorithms.

I. Model and results

Consider a set of r source packets which are multicast using distributed randomized network coding in finite field $\mathbb{F}_q$. Let the data content of each packet be represented by θ symbols $x_1, \ldots, x_\theta \in \mathbb{F}_q$, from which $\phi \leq \theta$ hash symbols $y_1, \ldots, y_\phi$ are calculated. We define the function $\pi : \mathbb{F}_q^h \rightarrow \mathbb{F}_q$ mapping $(x_1, \ldots, x_h) \in \mathbb{F}_q^h$ to $\pi(x_1, \ldots, x_h) = x_1^2 + \ldots + x_h^{h+1}$, and set

$$\begin{aligned} y_i &= \pi(x_{(i-1)k+1}, \ldots, x_{ik}) \quad \text{for} \quad i = 1, \ldots, \phi - 1 \\ y_\phi &= \pi(x_{(\phi-1)k+1}, \ldots, x_\theta) \end{aligned}$$

where $k = \left\lceil \frac{\theta}{\phi} \right\rceil$ is a design parameter inversely related with communication overhead. Let M be the matrix whose i^{th} row is the concatenation of the data and corresponding hash value for source packet i. A genuine packet contains a random linear combination of one or more rows of M, along with the vector of coefficients of the combination.

Consider a set of s good packets and $r-s$ modified packets being used for decoding. The good packets can be represented by matrix $C_a\,[M|I]$, where the i^{th} row of C_a is the coefficient vector of the i^{th} packet. The modified packets may contain arbitrary data and hash values, and can be represented by $[C_b M + V|C_b]$, where V is an arbitrary $(r-s)\times(\theta+\phi)$ matrix.

Theorem 1 *The attacker cannot determine which of a set of $q^{s\times rank(V)}$ potential decoding outcomes the receiver will obtain. For each of s or more packets, the decoded value will be one of $q^{rank(V)}$ possibilities $\{\underline{m}_i + \sum_{j=1}^{rank(V)} \gamma_{i,j}\underline{v}_j | \gamma_{i,j} \in \mathbb{F}_q\}$, where vectors $\underline{m}_i, \underline{v}_j \in \mathbb{F}_q^{\theta+\phi}$ are known to the attacker.*

Theorem 2 *The decoded packets can be consistent under at most a fraction $\left(\frac{k+1}{q}\right)^s$ of potential values of the good packets; at most a fraction $\left(\frac{k+1}{q}\right)^s$ of potential outcomes can be consistent. If the receiver decodes from multiple sets containing s' good packets in total, then this fraction becomes $\left(\frac{k+1}{q}\right)^{s'}$.*

This result explicitly characterizes the relation between detection probability, communication overhead ($\frac{1}{k+1}$), network code complexity q, and the number of good packets s', which may be viewed as a measure of the time taken to detect an attack or an inverse measure of the seriousness of the attack.

[1]This research is supported in part by NSF Grants CCR-0325324, CCR-0220039 and CCR-0325496, University of Illinois subaward #03-25673, Hewlett-Packard 008542-008, and Caltech's Lee Center for Advanced Networking.

References

[1] T. Ho, R. Koetter, M. Médard, D. R. Karger and M. Effros, "The benefits of coding over routing in a randomized setting", ISIT 2003.

[2] T. Ho, M. Médard, J. Shi, M. Effros and D. R. Karger, "On randomized network coding", Allerton 2003.

[3] P. A. Chou, Y. Wu and K. Jain, "Practical network coding", Allerton 2003.

[4] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance", OSDI 1999.

[5] K. P. Kihlstrom, L. E. Moser and P. M. Melliar-Smith, "The SecureRing Protocols for Securing Group Communication", HICSS 98.