Limiting Duplicate Identities in Distributed Systems

(A position paper)

Elliot Jaffe, Dahlia Malkhi, Elan Pavlov
The Hebrew University of Jerusalem

ABSTRACT

We explore a form of attack on a distributed system in which one or more nodes in the system maintain multiple identities. We argue that this attack is endemic to non-centralized systems. We present a number of defenses along with their limitations.

1. INTRODUCTION

An underlying requirement in many survivable systems is an upper bound on the fraction of corrupted participants. For example, in many distributed algorithms, a threshold of a third is a known resilience upper bound. In other settings, for example, a routing mesh like a hypercube, if corrupted nodes are as few as a tenth of the nodes, they already control most of the routing paths (whose length is logarithmic) for sufficiently large system sizes. Therefore, the resilience threshold is lower.

If an attacker is able to physically infiltrate a distributed system with many corrupt participants, more than the resilence threshold with which the system was designed, it can compromise any protection built in the system. It is therefore assumed (also in this work) that this is not the case, i.e., that the fraction of faulty participants in the systems conforms with some assumed upper bound.

However, this may still leave the system vulnerable. In an attack recently named Sybil [4], an attacker can assume multiple identities and, although physically it does not pass the assumed resilience bounds, in practice it plays the roles of more participants than the assumed threshold. This may again lead to a complete compromise of system safety.

In this position paper, we focus solely on the problem of enforcing a bound on the number of identities assumed by an attacker in a distributed system, under the assumption that its physical strength is appropriately limited.

It has often been suggested to use computational power in order to limit masquerading over the Internet. For example, Microsoft's anti-spam project [1] revolves around an old idea by Dwork and Naor [5], offering to use computation as 'stamps' for Email. Computational challenge was suggested by Franklin and Malkhi in [6] to perform web page usage metering. This seems like a good idea, but in a peer environment with no centralized control, it is not obvious how to use it. Some of the issues that need to be addressed are:

- Who would present and verify the computational challenges to participants? We should be careful not to let the attacker "vouch for itself", simply by having different identities vouch for one another.
- How will the verifier certify valid participants? In a peer network, it is unreasonable to assume a central certification authority. This seems to completely rule out the possibility that a peer will possess a certificate of honesty that it can directly present to other participants.
- How to prevent a slow infiltration of an attacker that gradually assumes more and more identities? There needs to be a fine tuning that allows, on the one hand, legitimate users to do useful work, while on the other hand, prevents the attacker from spending all of its time just assuming identities.

In this work, we focus on the problem of limiting an adversary to a bounded number of identities in a given time period. That is, an adversary may interact with one or more good nodes over a specified time period using a bounded number of identities.

The paper explores our on-going effort towards a practical protection of distributed systems against the Sybil attack. It describes a number of practical protocols for controlling the number of false identities assumed by its participants using reasonable assumptions about an attacker's power. We present both the strengths and the limitations of the protocols, and provide guidelines towards a future robust solution, that can be practically implemented and quantified.

1.1 Vulnerable Systems

Any system that depends on data sharing amongst a fixed number of players is potentially vulnerable to a Sybil attack. In this section we describe three such systems that are currently active research topics.

Distributed storage systems such as [2],[12] replicate encrypted data across a set of shares in order to gain resilience in the face of node failure and in order to limit the ability of any one node to gather enough information to uncover the plain-text data. The basic mechanism is to spread the data across a set of random nodes. If a Sybil node can maintain an unlimited number of virtual identities, then at some point, the system will unintentionally deliver sufficient data slices to a single physical attacker for that attacker to decrypt the data.

Overlay networks define an addressing and routing mechanism that is orthogonal to the underlying networking system. The benefits of such networks include the ability to route data between nodes using non-compatible networking system, content addressable networks and anonymity. Recent works [11] have explored the robustness of overlay network systems to failures of internal nodes. The problem is whether such systems can continue to route messages through alternate paths. The resiliency of these systems is measured by the number of distinct routes between any two nodes. An attacker can masquerade as a sufficient number of routing nodes such that all paths to the destination node pass through one or more of its false identities. In such a case, the attacker can perform a man-in-the-middle attack without the knowledge of either the sending or destination nodes.

Recently, Rabin has proposed a Virtual Satellite model [10], now being implemented at Harvard, which extends the bounded storage model of Maurer [8] for unbreakable encryption [3, 9] to practical peer to peer systems. In this model, each peer locally generates random pages which are replaced very frequently. Two nodes use their shared key to select which peers in the network to take pages from, and subsequently, the pages are replaced. Since an adversary cannot access all peers in a short time, it cannot access all such pages and thus the one-time pad is unbreakable. If an adversary can maintain a sufficient number of multiple identities then it is possible for that attacker to be the node(s) which provide the random pages, thus allowing the attacker access to the random data and hence significantly weakening the resulting one-time pad.

2. SIMPLE APPROACHES

We explored a number of sub-optimal approaches in order gain an understanding of the problem space. In this section, we describe two obvious approaches and detail some of their limitations.

2.1 Symmetric Verification

Assume that two nodes to communicate with each other. In order to inter-operate, each node cryptographically challenges the other at a regular interval. We assume that the cost of generating a challenge is negligible.

In this protocol, the number of identities that either node may maintain is strictly limited. We assume that a node can perform a bounded number of operations per time unit. In this protocol, each additional communications partner adds a fixed computational overhead. The limit is reached once the node no longer has resources with which to respond to new challenges.

The drawback of this protocol is that there is a fixed limit to the number of nodes that any given node can communicate with. This approach may be very reasonable for a Distributed Hash Table system where the degree of any node is small. It is unreasonable for a system where the in or out degree of any given node is significant. For example, in Client/Server systems the in-degree of a server is potentially the number of clients in the network. In Distributed Storage systems, the opposite situation occurs. A client node wishes to storage data on as many remote nodes as possible in order to increase the chances of eventually retrieving the data. In this case, the out-degree of the client node may be in the hundreds of nodes.

2.2 Probabilistic Approach

Assume that at every time period, each node in the system randomly chooses another node to challenge. Each node is challenged on average once each time period. A Sybil node maintaining N identities would then be challenged on average N times each time period. Since each node has a limited number of resources, a Sybil node would be unable to maintain an unlimited number of identities.

There are two problems with this approach. The first is that on average, a node is challenged once per time period, with high probability some good nodes will be challenged a logarithmic number of times (in the system size), hence, they will fail the challenge due to limited resources. Thus in this approach, even in a system with no malicious nodes, a few good nodes have a reasonable chance of being kicked out of the system during each cycle.

Worse yet, malicious nodes can utilize this system to kick out good nodes at will. Since malicious nodes are not restricted to choosing challenges randomly, they may collude to challenge any given node or set of nodes. These target nodes can then be trivially overloaded.

An additional problem with this approach is how to distribute the information about the failed nodes. Any simple mechanism can also be used by malicious nodes to arbitrarily eject good nodes from the system.

3. THE SISYPHUS PROTOCOL

We believe that any reasonable approach is going to depend on cryptographic challenges. We have begun to develop a protocol which shows some progress in addressing the Sybil attack.

We call this protocol the Sisyphus defense in reference to the character in Greek legend who was doomed to ceaselessly roll a rock to the top of a mountain only to have the rock roll back down of its own weight. In our protocol each node may interact with the system only as long as it is regularly performing cryptographic challenges. If a node stops responding to challenges, its identity is no longer acceptable.

Our system model is that of a set of connected physical entities or nodes. We assume that messages have a finite transmission delay and can be discarded after this period. This is necessary to bound the time that a node has to respond to the challenge. Without it, a node could collect and calculate the challenges, but delay responding to the challenge. It would thus seem that all the challenges were performed simultaneously, even though the node did not have resources to accomplish this feat.

Nodes may have one of two roles; Player or Voucher. When a Player enters the system, it selects an identity based on its address. We assume that some portion of the bits in its address are defined by the network itself. Addresses outside of this range will not be routed to this node. This assumption is true in the Internet and in Ah-Hoc networks.

Once the Player has decided on an identity, it repeatedly executes a well-known hash algorithm to generate a bounded list of K random identities. This set of identities defines a group of Vouchers for that Player. Given a node's identity, any node within the system can re-generate this exact set of Vouchers.

Upon entry to the system, the Player sequentially contacts each member of its own Vouching Node Set and requests a cryptographic challenge. The Player performs the challenge and returns it to that Voucher. The Vouching node can verify the result in linear time. The successful challenge is then accepted by the Voucher as an identity certificate for this Player, along with a time-to-live value. At this point the Player can initiate communications with other Players in the system.

A Voucher is responsible for maintaining the valid certificates and time-to-live of each node for which it vouches. Around the time that the certificate expires, the Voucher should re-challenge the node and record the results.

When a Player attempts to communicate with another Player, both Players contact each other's Vouchers and request verification of the other's identity. Each Voucher responds with a true or false value and with the remaining time for which it is valid. The identity of the remote Player is accepted by the local Player if a majority of the Vouchers accept the remote Player's identity. The identity is valid only for the time remaining on the oldest certificate. Once this time expires, the Player must recheck the results with the Vouchers.

We claim that because a Player controls only a limited number of bits in his address, the uniformity of the hashing function will generate a random set of Vouchers. It can be shown that the vouchers set size K can be fixed such that with high probability, an adversary cannot become its own Voucher. The proof of this is not presented here due to space limitations. Thus we may trust the Vouchers to perform the challenge service for each Player.

4. CHALLENGES AND FUTURE DIRECTIONS

There are a number of practical challenges to defending against a Sybil attack. We are exploring the following issues.

4.1 Routing

Our protocol employs a hash function to generate the addresses of Vouchers. The function generates values over a large range. It is highly unlikely that there exist in the system nodes with exactly these addresses. We must therefore assume that the system provides some routing function that maps a hash value to an existing node, whose identity most closely matches the hash value. This functionality is exactly that provided by a Distributed Hash Table (DHT) such as Chord [14]. Unfortunately, most DHTs designs mandate that nodes have very low in and out degrees, hence logarithmic routing paths. The logarithmic path length allows an attacker to corrupt a significant portion of the system with very little effort, by controlling many paths. Once a message routes through a node controlled by an adversary, it could route it to known malicious nodes or handle the messages itself instead of forwarding the message to the correct node.

One approach that circumvents this problem is to utilize a routing mechanism that has a high node degree, hence sufficient redundancy in routes, e.g., as in [7, 13]. An alternative approach is to utilize a small core of (mostly) trusted nodes in order to maintain a map of existing identities. In this way each node as it enters the system can perform a Sisyphus join with the trusted nodes. Since these nodes are trusted, the join need not be symmetric. Only the joining node needs to verify itself. The trusted nodes can maintain a list of all nodes in the system which can then be selected randomly indexed by the hash function.

A practical problem with the trusted core approach is its maintenance over time. Recent research in reconfigurable dynamic byzantine quorum systems [13] suggests that it is possible to build a trusted core that can be extended and reliable even as nodes enter and leave the trusted core. This could lead to a practical implementation of the Sisyphus protocol.

4.2 Tuning and Performance

We believe that the Sisyphus algorithm is not only theoretically possible, but also practical. We plan to explore the trade-offs between the cost of the cryptographic challenge and the ability of a node to perform useful work. Practical values for the number of Vouchers is also relevant, as larger number of Vouchers increases the reliability of the vouching set at a cost of additional challenges performed by each Player.

4.3 Denial of Service

Finally, distributed systems are plagued by Denial of Service (DOS) attacks. We plan to investigate effective ways to leverage the Sisyphus defence in order to protect against DOS attacks. Having enforced a strong correspondence between identities and real resource, we hope to limit DOS attacks by bounding the amount of resources servicing any particular identity.

5. REFERENCES

[1] M. Abadi, A. Birrell, M. Burrows, F. Dabek, , and T. Wobber. Bankable postage for network services. In *Proceedings of the 8th Asian Computing Science Conference, Mumbai, India*, Dec 2003.

[2] A. Adya, W. J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer, and R. P. Wattenhofer. FARSITE: Federated, available, and reliable storage for an incompletely trusted environment. In *Proceedings of the Fourth Symposium on Operating Systems Design and Implementation*, Dec. 2002.

[3] Y. Z. Ding and M. O. Rabin. Hyper-encryption and everlasting security. In *19th Annual Symposium on Theoretical Aspects of Computer Science*, volume LNCS 2285. Springer-Verlag, 2002.

[4] J. Douceur. The sybil attack, 2002.

[5] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In *Lecure Notes on Computer Science 740 (Proceedings of CRYPTO'92), pp. 137-147*, 1993.

[6] M. K. Franklin and D. Malkhi. Auditable metering with lighweight security. In *Journal of Computer Security 6, pp. 237-256*, 1998.

[7] I. Gupta, K. Birman, P. Linga, A. Demers, and R. van Renesse. Kelips: Building an efficient and stable P2P DHT through increased memory and background overhead. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, 2003.

[8] U. M. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

[9] M. O. Rabin. Hyper encryption and everlasting secrets: A survey. In *CIAC: Italian Conference on Algorithms and Complexity*, 2003.

[10] M. O. Rabin. Hyper-encryption and provably everlasting secrecy. Presentation at the Hebrew University, March 2004.

[11] S. Ratnasamy, S. Shenker, and I. Stoica. Routing algorithms for dhts: Some open questions. 2002.

[12] S. Rhea, C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon, and J. Kubiatowicz. Maintenance-free global data storage. *IEEE Internet Computing*, 5(5):40–49, 2001.

[13] R. Rodrigues and B. Liskov. Rosebud: A Scalable Byzantine-Fault-Tolerant Storage Architecture. Technical Report MIT-LCS-TR-932, MIT Laboratory for Computer Science, 2004.

[14] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM Press, 2001.