Rethinking Datacenter Addressing

Today's datacenters use Ethernet and the Internet Protocol to address nodes with static *destination* addresses. This makes them insecure, brittle and hard to manage. Future datacenters will be Zero Touch, dynamic, and use *relative* addressing, where routes follow *itinerant* VM's, containers or lambdas.

Instead of the Any-to-Any (A2A) addressing mode used in the open internet, the Earth Computing design uses Neighbor-to-Neighbor (N2N) addressing, more suitable to the dense, private, and *temporally intimate* communications between servers within rows of racks in a datacenter.

Another issue with conventional shared networks is that they employ the *source-destination* mode of addressing, where addresses are tied to *physical* devices. This creates a premature distinction between *identity* and *individuality*, which impedes the implementation of fault tolerance (failover), replica sharing (load balancing), and replica migration (mobility). Additional mechanisms (palliatives) must now thus be deployed on top of conventional networking to 'undo' this premature distinction (NATs, VIPs and MAC spoofing).

The reason destination indistinguishability should be considered a primary requirement in coherent networks is because modern application infrastructures require highly dynamic expansion and contraction of the replica sets (automatically triggered by requests). Mobility is at the heart of being able to easily migrate VM's and containers from one place to another; failover is at the heart of fault tolerance; load-balancing is at the heart of scalability.

Address-Free routing overcomes all these difficulties, by making all addressing relative. The 'direction' (port) of a cell now carries the local state enabling decisions on whether one or more substitutable replicas exist down that branch, or if a particular branch contains an exclusive token, such as an ownership (or exclusive) state, or a serialization token.

As an analogy, we achieve indirection in our postal system by notifying the local post office when we move. This notification (at some cost, and only for a limited duration) enables the local post office to redirect traffic to the new address while the owner notifies all possible senders of mail of their new address. The failure modes in this are obvious. This imposes additional work on the post office, the owner can never be sure they have a comprehensive set of all possible senders, and the owner perhaps doesn't want all senders to be able to automatically redirect their junk mail to them[1].

Container Addressing

Managing IP addressing has always been a complex, error-prone and semi-static process in datacenters. When IP addresses remained static for long periods of time, this was less of a problem. In today's

N2N (address-free) makes it easy to make purely local decisions on routing. This provides an an important benefit to management simplicity, which can somewhat be achieved without AIT.

What we get out of the combination of N2N and AIT is failover trees: we can just start using the alternate link for failover, and I know that you know that I will do that when we both see the link has failed. We don't have to go through an extensive re-initialization protocol (such as Dijkstra) to begin the re-routing. When links fail independently we can quickly and deterministically shift to the next preallocated route for any tree.

Key issue (address redirection): when a mobile entity, such as VM, container, or file, moves within the datacenter, the addressing should move with it automatically. Identifiers for these kind of objects are singular – they identify the object – not the physical node they happen to reside on right now.

Conclusion: The consequence of the Address-Free routing method is that it frees up the replica-sets to expand, contract, and float freely throughout an infrastructure on-demand, without having to change their destination addresses each time they (or their properties, such as ownership) move. Although this addressing mode can be programmed into current NICs, this is incompatible with current switches because their hardware (the chips themselves) are designed to do learning based on source-destination pairs. This means that many more packets now need to be treated as an exception and handled in software, which negates the performance advantage of having switches in the first place[2].

Business: The Importance Of Solving These Issues

These technical issues addressed above are important because, for anyone who can solve them, they address fundamental issues in datacenter design and internet services and open up new business opportunities by enabling the construction of on-line services which can:

- Enable the design of reliable file and object synchronization protocols where the corner cases disappear.
- Provide alternative recovery options, leading to Infrastructures which are much more robust to perturbations (failures, disasters, attacks) particularly as they grow in scale and traffic load.
- Enable self-healing systems which require less administration, and thus less opportunity for human error (the biggest impediment to resilient, secure and agile infrastructures today).
- Resolve fundamental issues regarding data synchronization that plague all internet services (Amazon, Apple, Dropbox, Google, Microsoft, etc.).
- Distribute transactions far faster, more reliably and more securely than existing systems.
- Enable core mechanisms for a secure systems foundation for resilient infrastructures. E.g., it leads the way to system defenses which can be compromised only if the attacker defeats the speed of light.

Product

shared networks: the last vestige of the monolithic mindset. Our product is a new datacenter fabric: the network defined software that creates and manages node, container and microservice relationships.

Our business plan shows how this can be leveraged into some key first mover products for datacenters today. We plan to go to market in three phases: (I) Control Plane only. (II) Configuration Data Plane, and (III) Data Dissemination Plane.

Market

Most current products in the market are related to threat detection, often after the fact, with increasingly complex behavior anomaly detection. Most security teams still rely on multiple point products for threat detection. According to ESG, it's time for a new, strategic approach. The combination of RAFE, TRAPHs and AIT provides the tools to create a strategic foundation for cyberdefence.

By 2020 cloud computing will be a $191B market, Amazon reported $2.41B in revenue for AWS division during the fourth quarter; $9.6B in annualized sales. [Forrester]. Dropbox took 2-1/2 years to build their a customized alternative to Amazon. Now Installing 40-50 racks/day each with 8 machines

Potential customers include: Comcast, Verizon, AT&T: "Temporal Intimacy" Applications. Low latency, Zero Touch, Zero Time to Recover, Zero Trust.

Existing Byzantine-resilient replication protocols satisfy two standard correctness criteria, safety and liveness, in the presence of Byzantine faults. In practice, however, faulty processors can, in some protocols, significantly degrade performance by causing the system to make progress at an extremely slow rate. While correct in the traditional sense, systems vulnerable to such performance degradation are of limited practical use in adversarial environments.

RAFE lets us eliminate ACL's, incorporate capabilities and eliminate entire classes of vulnerabilities due to the network.

Notes

[1]DNS is an example of how to separate naming from addressing. See the section 'The trouble with DNS'.

[2]We anticipate being able to reprogram current NIC's for the *address-free* mode of communication, but not current switch hardware. However, this is not to say that modifications to that hardware wouldn't be able to achieve what we need; these modifications may even be minor. However, this represents an additional go-to-market risk, and brings us right back to the question of why not just merge switches and servers into swervers (what we call `cells`).