Journal of Computer Science 9 (12): 1680-1685, 2013
ISSN: 1549-3636

doi:10.3844/jcssp.2013.1680.1685 Published Online 9 (12) 2013 (http://www.thescipub.com/jcs.toc)

THE RATIONAL BEHIND THE SERVAL NETWORK LAYER FOR RESILIENT COMMUNICATIONS

[1]Paul Gardner-Stephen, [2]Andrew Bettison, [1]Romana Challans and [2]Jeremy Lakeman

[1]School of Computer Science, Engineering and Mathematics,
Faculty of Science and Engineering, Flinders University, Adelaide, Australia
[2]Serval Project Inc., Adelaide, Australia

Received 2013-10-24, Revised 2013-11-01; Accepted 2013-11-18

ABSTRACT

The Internet Protocol is the dominant network protocol used in public networks today and has proven to be highly effective for wired networks and wireless networks alike, provided network address allocation can be coordinated. Mesh networks consisting of highly mobile devices present new challenges, especially when the assumption of coordination does not apply. One situation where coordination is not readily possible is ad-hoc networks in isolated areas and in disaster zones, both of which are characterized by deprivation of infrastructure. This study describes our realizations of several problems that IPv4 and IPv6 networking faces in such contexts and provides a brief description of an alternative network architecture for such situations, the Serval Network Layer and provides some of the reasoning behind the design decisions made. The Serval Network Layer is implemented as an open-source user-space network layer with strong intrinsic security characteristics and is able to be deployed without any centralized coordination.

Keywords: Network Layer, Layer Three, MANET, Mesh Networking, Serval Mesh, Overlay Networks

1. INTRODUCTION

The Serval Mesh (Gardner-Stephen, 2010; 2011; Gardner-Stephen *et al*., 2012; 2013) is software for mobile phones and other devices to form secure, self-organising and fully distributed mesh networks. These networks are intended to enable people to keep connected during disasters, as well as to support the social and economic resilience and growth of people living in isolated areas, low-income contexts and other environments where reliance on cellular infrastructure is unavailable, unaffordable or unwise. An early version of the software has been trialed in Nigeria (ICIL, 2012).

In contrast to most mesh networking initiatives (Anitha and Chandrasekar, 2011), the Serval Project has elected to implement a custom network layer rather than use IPv4 or IPv6. This unorthodox approach has drawn significant attention, including through an international challenge to prevent atrocities (HU and USAID, 2013) and a global security technology competition (Innocentive, 2013). This study seeks to explain the motivations for this unorthodox approach and the benefits that it brings to the Serval Mesh software. It is hoped that this will spur healthy discussion about this decision to the benefit of the mesh networking community. The Serval Network Layer is implemented in the Serval Mesh software, the source code of which can be downloaded from (Gardner-Stephen *et al*., 2013), allowing examination and experimentation by third parties. The compiled Serval Mesh software can be downloaded from the Google Play store and run on devices running Android 2.2 or newer.

2. COMMUNICATING DURING DISASTERS

Most infrastructure and mesh networks are designed with peace-time assumptions, in particular, that devices on the network are able to access some sort of coordinating centre, typically via the internet, or in some

Corresponding Author: Paul Gardner-Stephen, School of Computer Science, Engineering and Mathematics,
Faculty of Science and Engineering, Flinders University, Adelaide, Australia

cases, in the local community itself. This allows the network to be deployed, managed, monitored and operated in an efficient and effective manner.

For example, cellular networks require registration with the network operator to enable a device to participate in the network. The network allocates the device's identity on the network, in this case a telephone number. Another common example is the use of DHCP and other IP address allocation schemes on many types of network.

Without such a coordinating centre, these networks are not possible to deploy. For example, without a functional network operator a cellular network cannot admit new devices and in most instances cannot carry any traffic. Similarly on an IP network, without a DHCP server or similar facility, new leases on IP addresses cannot be offered and existing IP address leases will eventually expire.

In some cases, such as IP networks, it is possible to self-allocate an address. This is a non-trivial undertaking and lends itself to address collisions and a variety of other configuration and interoperability failure modes that are undesirable.

Unfortunately, during a disaster, the availability of the network coordinating centre cannot be assured. Yet this is often a time when it is vital to admit new devices to the network. For example, following the Great Haiti Earthquake of 2010, while partial cellular service was maintained, it is unlikely that the local carriers were able to register many new SIM cards for the influx of humani-tarian aid workers.

This is one of several challenges that was faced by the Serval Project in its objective of making it possible not only to maintain communications during a disaster, but to actually roll out a network I critically infrastructure-denied disaster zones so that people can communicate with one another, coordinate their responses, help to maintain rule of law and ultimately, to help their communities to suffer less damage, as well as to recover faster and more completely.

3. IP ILL-SUITED FOR FULLY-DISTRIBUTED, INFRASTRUCTURE DEPRIVED MOBILE NETWORKS

The Serval Mesh software was designed with the disaster use-case in mind and consequentially, a prominent design criterion was that it must be able to operate without any sense of a network coordinating centre or service. Practically, this means that devices must be able to self-admit to the network, which entails self-allocating a network address and without any realistic probability of causing a network address collision. It is also highly desirable that communications be able to be secured, while keeping the software trivially easy to deploy and operate.

The need for self-allocation of addresses poses difficulties for a traditional IPv4 network. While in principle it is possible to define a large network space, such as the 10.0.0.0/8 space, the probability of address collisions is non-trivial. Despite that space containing 2^{24} network addresses, due to the Birthday Paradox after only a few thousand nodes it becomes more likely than not that there will be at least one address claimed by more than one node. This entails that nodes must either have a separate identity from their IP address, so that collisions can be resolved while the network is operating, or that the network size must be constrained.

If we take the technology to the logical limit of al-lowing the global population to participate in the mesh and enjoy free mobility, then IPv6 becomes strained, as there are $>2^{32}$ people in the world and according to the Birthday Paradox we need $>2^{64}$ host addresses to en-sure safe self-allocation. The situation becomes worse if we consider the development of the Internet of Things, where the total global device count may well exceed 2^{40} or even 2^{48} devices at some point. In that case 2^{80} to 2^{96} host addresses are required. Yet due to the address structure of IPv6 there are only 2^{64} host addresses available, while incurring a 128-bit address overhead. At best IPv6 offers poor value and at worst it is illsuited to the task.

Even assuming that IPv4 or IPv6 could solve the address allocation problem in a dynamic and totally decentralised network, security is left unaddressed. IP addresses are not axiomatically tied to an identity. That is, some method of associating IP addresses and identities is required, such as IPSec. Unfortunately, IPSec is far from trivial to deploy and has significant limitations when deployed in an environment that lacks a coordinating centre that includes a source of authority, such as a certificate signing authority. Yet private correspondence is a critical need in disaster communications. For example, personal data privacy laws are not suspended in most jurisdictions and responders may need to communicate medical information of victims.

Without a secure means of transport, communications of such information has significant risk of breaching privacy laws as well as the natural justice right to privacy of individuals and therefore the need for secure correspondence cannot be ignored.

Even assuming that IPv4 or IPv6 and IPSec posed a possible solution, for ad-hoc disaster response networks these have a critical problem: not all devices include support for these protocols, especially IPSec. Indeed, on many devices the kind of network management activities to enable IPv4 or IPv6 to operate in a totally infrastructure-denied environment is not possible without modification of the operating system, which typically requires access to some sort of centralised service to accomplish.

In short, there are considerable challenges to using the traditional IP-based networking approach for our target use case of enabling communications during the acute phase of disasters. This led us to consider non-IP based solutions and in particular a custom network layer implemented in user-space so as to avoid the need for administrative privilege on the host operating system, since such access is an unattractive proposition on mobile telephones and other devices, assuming that it is even possible.

This user-space approach maximises the range of devices that we can support and allows the resulting software to be more portable, because there is no need to integrate with each additional operating system. Instead, all that is required is access to UDP or ethernet sockets to allow tunnelling of the Serval Network Layer over existing IP transports, such as Wi-Fi. Moreover, using a user-space implementation allows for the rapid development and integration of non-traditional network transports, such as low-bandwidth packet radio interfaces, which can then be made to run the Serval Network Layer natively, without tunnelling over IP. This is the method used to add UHF packet radio support on the prototype Serval Mesh Extenders. The user-space approach also considerably simplifies simulation and testing, because the network can be virtualised beneath a number of Serval Mesh processes and as a result the automated tests of the Serval Mesh source code include a number of network topology tests that can be run in seconds to minutes and that would ordinarily require the use of complex simulation machinery of some sort. The performance impact of the network layer being in user-space is discussed in a later section.

4. AN ALTERNATIVE APPROACH: MERGING PROVABLE NODE IDENTITY AND NETWORK ADDRESS

The fundamental challenges outlined above are establishing a network layer that has large enough address space to allow potentially hundreds of billions of devices to safely self-allocate network addresses to facilitate self-admittance to the network and providing a simple and effective mechanism for securing communications between any pair of nodes on the network, all without any recourse to any coordinating centre or authority. Ideally the resulting network layer will be more bandwidth efficient than IPv6 to help conserve the limited bandwidth available on typical mesh networks.

The approach taken for the Serval Mesh was the creation of the Serval Network Layer (SNL). The Serval Network Layer uses 256-bit Elliptic Curve public keys as the primary network address and combines this is an effective address abbreviation scheme that reduces network overhead to IPv4-like levels, despite offering 2^{192} times more host addresses than IPv6.

Elliptic Curve Cryptography (ECC) has several advantages over the more common RSA and related cryptographic systems. First, in well designed ECC systems there are inconsequentially few bad keys. That is, a 256bit key space contains approximately 2^{256} usable keys and so a compact key space is usable. In contrast, systems that operate by multiplying two (hopefully) prime numbers to produce a public key have a key density well below unity and even ignoring that problem, there continues to be a growing body of research that suggests that RSA and related dualprime systems are vulnerable to a terrifying variety of vulnerabilities. Indeed, the results of Lenstra *et al.* (2012) should give considerable pause to any new use of a dualprime based cryptographic system where substantial quantities of public keys can be easily collected.

A related benefit of well constructed ECC systems is that signatures and message authentication codes can be as small as 64 bytes, or even less in some ideal situations, allowing all packets to be authenticated, thus securely binding network addresses to identities and rendering spoofing of address impractical and hence obviating the need for more complex and probabilisitic approaches (Manjula and Chellappan, 2012).

Related to the above, because network addresses and public keys are unified, it becomes trivial to encrypt communications between pairs of nodes using a Diffie-Hellman shared secret calculation using the sending parties private key and the receiving parties public key. Consequentially, the Voice over Mesh Protocol (VoMP) created for the carriage of live voice calls over the Serval Network Layer is encrypted and authenticated by default. Techniques are still required to address Man-in-the-middle attacks, the protocols of which will be described in a future paper, along with the Voice over Mesh Protocol itself.

A further significant advantage of this approach is that secure communications can occur in a highly

partitioned network, because no key exchange or session information needs to be exchanged in order to create an encrypted or authenticated communication between two parties. This is a vital characteristic for disaster response and remote area networks where the node density may be too low to sustain a continuously connected network. This property has been used to implement an encrypted and authenticated text messaging protocol, MeshMS, that allows acknowledged delivery of text messages over highly partitioned networks. An early version of MeshMS is described in the work of Gardner-Stephen *et al*. (2012), but readers should note that the implementation has been completely overhauled since the publication of that study. An updated description of MeshMS will be the subject of a future paper.

Summarising the above, the result is that communications between parties on the network can be encrypted and authenticated, without recourse to any coordinating centre or authority, rather these features become intrinsic properties of the network itself. This is the approach taken by the Serval Network Layer that uses the Curve25519 ECC primitives from the NaCl cryptography library (Bernstein *et al*., 2012).

5. ABBREVIATING NETWORK ADDRESSES

As previously described, the Serval Network Layer uses 256-bit ECC public keys as network addresses. Naively, this would result in 768 bit address headers, 256 bits each for source, destination and next-hop. However, in many cases the next hop and destination are identical. Moreover, a single node is unlikely to have more than several hundred immediate neighbours and similarly is unlikely to be witness to particularly large numbers of flows. Therefore, intuitively, it should be possible to abbreviate network addresses. A simplified overview of how this is achieved in the Serval Mesh is described below. Full source code is available for those curious for the full details.

The Serval Network Layer network header format is made flexible to allow network addresses to be expressed either in full, or in one of several abbreviated formats. A sending node may elect to encode an address either in full or in abbreviated form. The receiving node when observing an abbreviated address either knows how to deduce the full address or not. If not, it sends a Hanson Packet to the sender requesting explanation of the abbreviation, to which the original sender, hopefully, responds. Abbreviations are re-encoded from the recipients perspective before being forwarded to the new next hop.

Abbreviated addresses can be as short as 8-bits using a combination of tokens and abbreviated addresses, which can result in a network layer header that is smaller than an IPv4 network layer header. As in many instances the next hop and destination address can be identical, there is a token that indicates that the destination matches the next-hop. In general, abbreviated addresses use as few bytes as possible whilst avoiding ambiguity. Because abbreviations are local to a specific hop and consist of the shortest prefix that uniquely identifies the address to the sending party, the abbreviations are most typically one or two bytes in length, offering substantial savings.

The overhead of the Hanson packets and responses must be taken into account in calculating the total overhead. Hanson packets contain the abbreviation to be explained, A Hanson response packet must contain the full 32 byte (256 bit) address being queried and in some cases must also include the full 32 bytes of the address of the party providing the explanation. A complete abbreviation resolution, including the various other protocol fields typically entails the transfer of between 50 and 150 bytes. Given that abbreviations are able to save 3×32 bytes -3×1 byte = 96-3 = 93 bytes per packet, this saving is typically recovered after just one or two packets. For packet streams, such as VoMP calls or Rhizome file transfers the overhead of abbreviation expansion is amortised to well below one byte per packet.

6. ON THE COMPUTATIONAL OVERHEAD OF THE SERVAL NETWORK LAYER AS A USER-SPACE NETWORK TRANSPORT

Most network layers are implemented in the kernel of the host operating system to avoid unnecessary memory copies between kernel and user space, so as to maximise performance. This is based on the high cost of switching between kernel and user space on modern processor architectures, typically due to Translation Look-aside Buffer (TLB) invalidation and cache pressure issues, with one study suggesting typical process-to-process context switching penalties of around 30 microseconds (Sigoure, 2010), which is concurred by another study showing the cost ranging from a few to a few hundred microseconds, depending on various parameters (Li *et al*., 2007).

On small mobile devices which are a primary focus of the Serval Mesh the story is a little different. First, the cache resources are often much smaller and so the

rela-tive degradation due to cache issues is reduced to some degree. In the case of processing network traffic, caches are of limited value, because the data is new each time and cannot be cached and so the playing field is substantially levelled.

Overall, the gap in performance between small embedded processors and "normal" processors is mostly in the number of computations that they can perform per unit time and much less in the number of kernel to user-space memory transfers that they can perform per unit time. To give an idea of the magnitude of this effect, consider the context switching speed of an Atheros 9331 (400MHz MIPS, 64KB instruction cache, 64KB data cache, running OpenWRT Linux) is compared with an Intel i7 (2.7GHz, ~4.5MB L2+L3 cache, running OSX 10.7.5). Despite the huge difference in computational performance, the Intel processor can perform only 3 to 4 times as many context switches per second (166,710 per second on the i7 versus 49,028 on the MIPS using the well-known context1.c pipe-based context switch benchmark).

Thus, while we agree with the general wisdom that unnecessary kernel to user-space context switches and memory transfers represent a cost that should be avoided wherever possible, the impact on mobile devices can be surprisingly slight. Thus, for our situation at least, we find against the common wisdom that user-space implementation of a network layer are too slow and indeed we find that performance is adequate. Indeed, in our empirical testing to date we find that the performance of our user-space network layer is acceptable, both on the 400MHz MIPS AR9331 processor and on low-end ARM processors found in sub-$100 Android smart-phones. Formal benchmarking of performance, while valuable, is beyond the scope of this study.

If performance does prove an issue on normal processors, such as the i7, then it is entirely feasible to explore a kernel-resident version of the network layer to completely obviate this concern. In other words, we have gained much and lost little by starting with a user-space implementation that may, or may not, end up in kernel-space on some operating systems in the future.

7. REGARDING INTEROPERABILITY WITH IP BASED APPLICATIONS

A concern that has been raised by a number of parties is that existing IP-based applications will not be able to make use of the Serval Mesh. These concerns, while understandable, do not detract from the utility and practicality of our system.

First, this is a temporary affair, as we have funding to implement a SOCKS over Serval Network Layer facility that will allow tunnelling of IP traffic over the Serval Network Layer. The full explanation of this will be the subject of a future paper, but in short, the SOCKS facil-ity will tunnel TCP and UDP connections using a custom Mesh Streaming Protocol (MSP), that will incorporate network coding methods similar to those described by Sundararajan *et al.* (2011), to offer substantially improved performance over lossy wireless links than using TCP natively.

Second, in many ways it is helpful for the operation of a mesh network for applications to have to opt-in to use the mesh. On the one hand, this is because it helps reserve bandwidth for critical mesh-enabled functions, such as disaster communications. On the other, it gives application writers the opportunity to consider the differences between relatively reliable internet networks compared with mobile wireless mesh networks. For example, the need for more robust methods for dealing with packet loss (as are being addressed in our SOCKS over MSP proxy), or more critically, the likelihood that the network will ordinarily be partitioned. This second issue requires rethinking data transport so that store-and-forward methods can be effectively leveraged (as in our MeshMS text messaging protocol).

8. CONCLUSION

We have provided some explanations for our unorthodox decisions that were made during the design of the Serval Mesh. We acknowledge that at this stage a number of the assumptions and arguments are based solely on empirical experience and warrant the collection of appropriate data that will enable us to at least quantify the impact of these decisions and guide future evolution of the Serval Mesh software. We look forward to any discussion that this exposition generates.

9. ACKNOWLEDGEMENT

We wish to acknowledge the support of The Awesome Foundation for the Arts and Sciences (Boston Chapter), Internews, The Shuttleworth Foundation, The NLnet Foundation, The New America Foundation, our numerous other individual and corporate donors as well as the assistance of the New Zealand Red Cross IT&T ERU and the open-source community.

10. REFERENCES

Anitha, P. and C. Chandrasekar, 2011. Comparative performance evaluation of routing algorithms in IEEE 802.15.4 and IEEE 802.11 with different ad hoc routing protocol. J. Comput. Sci., 7: 731-735. DOI: 10.3844/jcssp.2011.731.73

Bernstein, D., T. Lange and P. Schwabe, 2012. The security impact of a new cryptographic library. Proc. LatinCrypt Lecture Notes Comput. Sci., 7533: 159-176.

Gardner-Stephen, P., 2010. Serval DNA Source Code.

Gardner-Stephen, P., 2011. The Serval Project: Practical Wireless Ad-Hoc Mobile Telecommunications.

Gardner-Stephen, P., A. Bettison, D. Gardner-Stephen, R. Challans and J. Lakeman *et al*., 2013. Proceedings of the IEEE Global Human-Itarian Technology Conference, (TC'13).

Gardner-Stephen, P., J. Lakeman, R. Challans, C. Wallis and A. Stulman *et al*., 2012. MeshMS: Ad Hoc data transfer within mesh network. Int. J. Commun. Network Syst. Sci., 5: 498-504. DOI: 10.4236/ijcns.2012.58060

HU and USAID, 2013. Meet the tech challenge winners communicate.

ICIL, 2012. Mesh casting news in the Port Harcourt Waterfronts. Internews Center for Innovation and Learning.

Innocentive, 2013. Global Security Challenge 2013 Summit Summary and Winners Announced. Humanity United and USAID.

Lenstra, A., J. Hughes, M. Augier, J. Bos and T. Kleinjung *et al*., 2012. Ron was wrong, Whit is right. Cryptology ePrint Archive, Report 2012/064.

Li, C., C. Ding and K. Shen, 2007. Quantifying the cost of context switch. Proceedings of the Workshop on Experimental Computer Science, Jun. 13-14, ACM New York, USA.

Manjula, V. and C. Chellappan, 2012. Trust based node replication attack detection protocol for wireless sensor networks. J. Comput. Sci., 8: 1880-1888. DOI: 10.3844/jcssp.2012.1880.1888

Sigoure, B., 2010. How long does it take to make a context switch.

Sundararajan, J.K., D. Shah, M. Medard, S. Jakubczak and M. Mitzenmacher *et al*., 2011. Network coding meets TCP: Theory and Implementation. Proc. IEEE 99: 490-512. DOI: 10.1109/JPROC.2010.2093850