Exploring Trusted Networking for Protected Applications

D J Capelis James Larkby-Lahet Darrell D.E. Long

{djcapelis, jamesll, darrell}@cs.ucsc.edu
Department of Computer Science
University of California, Santa Cruz

Technical Report
8 June 2012
UCSC-WASP-12-01

Abstract

In modern computing systems, networking is critical. In the context of trusted application environments, building a trusted networking interface remains an open question. In this paper, we categorize the networking needs of trusted applications into three modes: *Local Equivalent*, *Trusted Local Networks* and *Internet-Wide Trusted Networking.* Collectively these modes enable trusted applications to do everything from securely accessing the network to establishing a valid identity on the local network, or even the Internet. We explore the hardware mechanisms available in currently shipping products which can be used to implement our interfaces. We discuss our plans to prototype these networking interfaces in our lab's trusted platform called *LockBox.* We conclude that our trusted networking design is feasible using existing hardware and is ready for implementation.

1 Introduction

Few groups have examined the needs of trusted computing platforms when it comes to providing networking functionality. Networking is often addressed after the core components of the system have been specified, implemented, and tested. Our trusted platform, *LockBox*, differs from many traditional trusted systems by allowing applications to maintain an environment with certain security properties that even the operating system managing the environment cannot violate. The networking challenges faced by trusted environments in systems like *LockBox* are different from most of the challenges examined by others.

The most prominent trusted networking work in a trusted computing context was developed by the Trusted Computing Group, (TCG) a broad consortium of almost every major technology company. Their work on the Trusted Platform Module (TPM), a hardware security module implemented in the platform chipset, was specified in late 2001 [8]. The goals and threat model of the TPM are different from the goals and threat model of *LockBox.* The TPM aims to protect the integrity of the software stack and *LockBox* aims to protect a user's data and applications from a malicious management stack. The networking specifications produced by the TCG, known as the Trusted Network Connect (TNC) series of standards, are similarly different from our own aims. The core of the TNC standards focused on protecting the network core from the edge, whereas our focus in *LockBox* is to drag the network edge further out and root it within the Trusted Application's protected environment and allow it to maintain a distinct network identity.

Historically, trusted networking work has lagged the trusted systems they were built to support by a number of years. The initial core Trusted Network Connect specification [9] came out in 2005, almost three years after the TPM specification was created and after the initial TPM hardware had begun shipping. The TNC standards continue to be an active area of effort within the Trusted Computing Group. For instance, a 2009 specification [10] was the first to outline how federated identity might be incorporated into the system. Adoption by traditional Internet standards bodies has taken several additional years. It wasn't until 2010 that the Internet Engineering Task Force (IETF) standardized a portion of the TNC specifications using the RFC process [7] [6].

In this Technical Report, we seek to sharply delineate the trusted networking issues we feel a system like *LockBox* will need to solve. We propose several security modes we think are critical for our threat model and we explain how those modes might be implemented. We specifically identify the hardware features which exist in modern networking cards that we expect to utilize to allow us to implement these security features with minimal performance overhead. We discuss the two environments we've designed to prototype these concepts. Finally, we conclude.

2 Defining Trusted Networking

In *LockBox*, we've designed a trusted computing architecture which creates protected contexts where a trusted application can maintain the confidentiality of its data even if the management software running between *LockBox* and the application is malicious. Our existing design has limited support for trusted channels and our existing effort has focused on utilizing trusted channels to allow a user in front of the machine to communicate with a trusted application in a trusted context using a keyboard. Expanding these channels to enable trusted applications to securely move data on to and off of a network from their trusted context is a key missing piece of our current design. To address this deficiency it is important to examine the specific security properties we wish to provide as well as acknowledge the ones we consider out of scope.

2.1 Access

Perhaps the most important property needed is access to the network which does not require passing sensitive data to management software running between *LockBox* and the end application. Existing systems require applications pass network requests to the operating system which multiplexes the network interface with its own networking driver. In the *LockBox* design, trusted channels provide direct channel to a networking interface. A trusted application can pass data directly with a virtualized network interface and does not rely on the operating system to multiplex the network interface.

2.2 Identity

The second security feature important to a trusted context is maintaining a notion of identity on the network. Being able to securely communicate on the network is of limited value if the network can't disambiguate data from a certain trusted context with from any other data flowing out the same interface. Enabling a trusted context to maintain an identity on the network is a key requirement for trusted networking.

There are two different types of identity we think are critical in a modern trusted system. The first is identity on the local network. This identity allows traffic from a trusted context to be disambiguated by the networking infrastructure from other traffic on the local network. The second is identity on the global Internet. This identity allows traffic from a trusted context to be disambiguated by other applications running on other computers anywhere on the Internet. Both types of identity are important to provide to trusted applications to enable a broad set of applications to use *LockBox*.

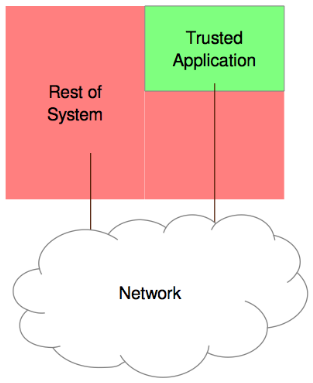
2.3 Out of Scope

Like the main *LockBox* architecture, our focus in the networking realm is on maintaining data confidentiality over availability. Since under the *LockBox* architecture the management software is allowed to destroy trusted contexts outright (but is required to do so in a certain way to maintain security properties), no network mechanism to ensure forward progress is necessary. An operating system may wedge the machine and trusted contexts would not be able to access the network. In addition, we do not propose a mechanism to allow a remote node to inspect state on the local machine. This is a feature commonly found in other trusted computing platforms, but those systems provide a very different set of security guarantees than *LockBox* provides. In *LockBox* the user plays a role in attestation and maintains control over the process. One surprising thing which is not in our scope is data payload privacy. Existing, proven technologies such as IPSec [4] and TLS [1] provide working, high performance and well deployed options to solve this issue. We did not see a need to reinvent this wheel.

3 Technical Design

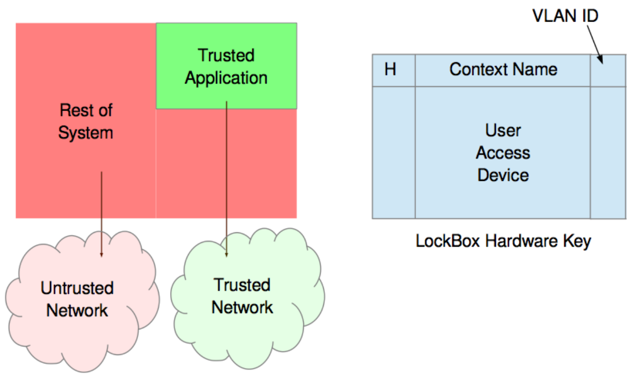
Our technical design splits the trusted networking needs for *LockBox* into three distinct modes. These modes can be combined with each other or used separately depending on the security needs and desires of the application. Some of these modes add additional storage requirements to the *LockBox* system and we will delineate the addition storage required as well as the security requirements for such data.

3.1 Local Equivalent



The first mode focuses on a baseline level of protection that allows a trusted context to communicate on the network without interference or snooping from the Operating System. Single Root I/O Virtualization (SR-IOV) can be used to multiplex a series of virtual interfaces on one network card out to trusted contexts and the host with minimal performance overhead. These interfaces can all be managed independently of each other and traffic sent and received by one interface can be separated from the others in the software stack. If the networking infrastructure and physical links are trusted, this alone can be enough to ensure some level of security for trusted applications which wish to interface on the local network. If they are not, existing network security technologies can be deployed.

3.2 Local Network Identity



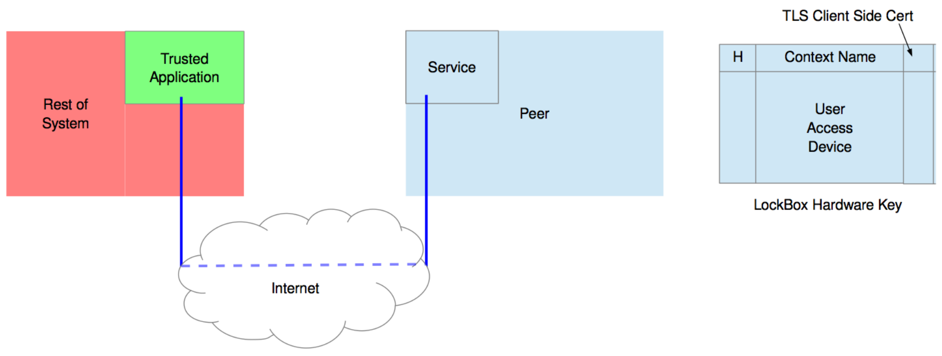
Providing an identity for trusted contexts on a network is simple step beyond providing Local Equivalence. The identity information comes from *LockBox*'s *User Access Device*, which is the piece of *LockBox* that already determines the name of each trusted context and the verification information required to check the code loaded into it. The user can include a port and VLAN tag and/or MAC address in the *User Access Device* for each trusted context the user wants to grant a local network identity to. Like other data held in the *User Access Device*, the security properties of our system rely on the data's integrity, but does not rely on its confidentiality. LockBox can then partition traffic from trusted contexts onto a separate physical trusted port (or several trusted ports, depending on the number of physical ports available on the network interface and the user's desire for separation) and tag traffic from trusted contexts with the VLAN tag and/or MAC address specified in the *User Access Device*.

Many modern network cards have the ability to restrict virtual interfaces so use of a specific VLAN tag or MAC source address can be specified before yielding control over the virtual interface [3]. Cards also include functionality to detect malicious drivers, which allows *LockBox* to prevent other virtual network interfaces running on the same trusted physical port from using an identifier which was not assigned. These features are embedded within the hardware and therefore should have minimal performance overhead.

These tags can be used by the hardware in the underlying networking infrastructure to make access control decisions. Since *LockBox* can dedicate a physical port exclusively to traffic from trusted contexts, the networking hardware can know that any traffic on that port with the appropriate VLAN or MAC source tags came from the trusted context associated with those tags. Traffic on other ports can remain unrestricted.

3.3 Internet-wide Identity

VLAN tags and MAC source addresses only provide an identity to the local network infrastructure. Not only is a flimsy form of identity, most interesting security applications span administrative domains and need identities that reach beyond the local network. For this, we utilize TLS client certificates [1] and like before, store identity information in *LockBox*'s *User Access Device*. Unlike other identity information stored in the *User Access Device* however, the confidentiality of this data



does impact the security constraints offered by the system. Only the trusted context assigned to the TLS client certificate should be able to read the private key material in the cert.

LockBox ensures that once a trusted context is properly initialized and verified, the client certificate assigned to the context can be loaded into its protected memory and remain confidential. The context can then use this certificate to open TLS connections across the Internet and verify its identity using its certificate. Recently a large amount of work has been put in to reduce the overhead of TLS as deployment of HTTPS has increased in several major web-based products. (Gmail, facebook and twitter are notable examples.) This overhead reduction combines hardware features like AES-NI [2], a series of hardware encryption acceleration instructions available on recent processors, with a range of software optimizations. Our use of TLS in this context provides a high performance and simple way to provide identities across the Internet.

4 Prototypes

Our prototypes for this trusted networking technology will use Intel's *i350-T2* network adapters, which provide our experimental environment with SR-IOV capable networking cards, each with two physical ports and a number of virtual interfaces and queues available on each port. These trusted network will be implemented in two very different environments to allow a broad understanding of how different types of operating systems and applications could use *LockBox* and these associated trusted networking features. XOmB [5] is an exokernel system which eschews compatibility and legacy to freely redesign OS abstractions and experiment with radical operating system change. In contrast, Linux runs millions of modern applications and traces its heritage back decades, in both codebase and concepts. Proving our networking interface concepts can work in both these contexts will make a strong case for general applicability.

4.1 XOmB

The *LockBox* framework designed for *XOmB* is targeted towards experimental work. *XOmB* is an experimental exokernel system which is an excellent match for experimenting with userspace networking stacks, userspace networking drivers and the higher level software interfaces which will be needed to make trusted applications in trusted contexts autonomous in their ability to directly communicate with a network card without the intervention of a traditional operating system kernel. This testing ground will be a good place to define and develop the initial architecture to easily allow userspace applications to run their own drivers and communicate directly with networking hardware.

4.2 Linux

Once the architecture is finalized in an exokernel, we intend to develop a Linux backend targeted towards existing production systems and prototyping real workloads that a system may run today using existing legacy applications. Our goal is to make a large amount of the userspace code independent of the Operating System and define only a small Operating System specific interface. The *LockBox* framework on this platform is designed to use existing virtualization technologies like KVM to implement *LockBox* features and can be used to prototype trusted networking features in a way which allows us get real performance data to define the overhead in our system.

5 Conclusion

The Trusted Networking features we propose to implement for the *LockBox* system rely on a different threat model than most previous attempts. These features are feasible to implement using existing hardware and are designed to re-use many of the newest hardware features created for virtualizable systems to achieve high levels of hardware support and performance. We propose a set of trusted networking features in userspace. Our prototypes will show trusted applications can feasibly use network cards in a userspace trusted channel without OS intervention or interference. Initially we plan to implement these features in an experimental research exokernel and then move them into Linux. This work will span two very different environments, allowing us to show our concepts generalize beyond any specific operating system. In summary, our design enables trusted contexts to access the network, use the network and promulgate an identity on either a local network or the global Internet.

6 Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1018928. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] T. Dierks and E. Rescorla, “The transport layer security (TLS) protocol version 1.2,” RFC 5246, Tech. Rep., August 2008.

[2] Intel Corporation. (2010) Intel Advanced Encryption Standard Instructions (AES-NI). [Online]. Available: http://software.intel.com/en-us/articles/intel-advanced-encryption-standard-instructions-aes-ni/

[3] Intel Corporation. (2012) Intel Ethernet Controller I350 Datasheet. [Online]. Available: http://www.intel.com/content/www/us/en/ethernet-controllers/ethernet-controller-i350-datasheet.html

[4] S. Kent, BBN Corporation, R. Atkinson, and @Home Network. (1998, November) RFC 2401. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2401.txt

[5] J. Larkby-Lahet, B. Madden, D. Wilkinson, and D. Mosse, “Xomb: an exokernel for modern 64-bit, multicore hardware,” in *WSO-VII Workshop de Sistemas Operacionais*, 2010.

[6] R. Sahita, Intel, S. Hanna, Juniper, R. Hurst, Microsoft, K. Narayan, and Cisco Systems. (2010, March) RFC 5793. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5793.txt

[7] P. Sangster, Symantec Corporation, K. Narayan, and Cisco Systems. (2010, March) RFC 5792. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5792.txt

[8] Trusted Computing Group, "TCPA Main Specification V1.0," *Trusted Computing Group*, 2001.

[9] Trusted Computing Group, "TNC IF-IMV Version 1.0, Revision 3," *Trusted Computing Group*, 2005.

[10] Trusted Computing Group, "Federated TNC Version 1.0, Revision 26," *Trusted Computing Group*, 2009.