PCPP: Private Computing on Public Platforms A New Paradigm in Public Computing

Thomas H. Morris and V.S.S. Nair
High Assurance Computing and Networking Lab (HACNet)
Southern Methodist University
Dallas, TX, USA
{tmorris, nair}@engr.smu.edu

***Abstract*-As cellular telephones and high capacity memory sticks emerge as users' primary repository for data and applications, users will often run applications and display data on remote hosts. The biggest challenge in supporting this mobile data, mobile applications, stationary platforms model is in ensuring security of the applications as well as the platforms. To that end, we propose PCPP as a new paradigm in public computing in which the remote host is not trusted, rather, security is owned and enforced by the application. PCPP is a two step process which ensures that the application running on the remote host remains unaltered, unmonitored, and unrecorded for future analysis by the public platform or any application running on the public platform.**

***Index Terms*- computer security, mobile code security, privacy**

I. Introduction

The world of nomadic and mobile computing is rapidly emerging. Part of this change is the increasing penetration and capability of cellular telephones. Cellular telephones are adding memory capacity and processor speed at a break neck pace. This trend combined with the cellular telephone's diminutive size lead many to believe the cell phone will emerge as users' primary repository for data.

One expectation is that users will hold their data on their cellular telephones but take advantage of available processors and displays in their immediate environment for display, file sharing, and processing power.

This computing model introduces new security challenges. A user must ensure that both the control flow and data flow of its application have remained unaltered, unmonitored, and unrecorded for future analysis for the period their application and or data reside on the foreign host. We see this as the defining challenge of Public Computing on a Private Platform (PCPP) [21].

PCPP is a new paradigm in nomadic computing security in which the user achieves security and privacy by identifying, attacking, and eliminating risk rather than by building a trust framework. We see eliminating risk and building a trust framework as distinctly separate approaches to security.

The PCPP approach is similar to the security provided to a state leader when traveling abroad. When a foreign dignitary travels abroad his or her security is also a two step process. First, an advance team is sent to scout the remote location. This advance team will work with local authorities to understand the security risks and safeguards that are already in place. If the advance team finds the risk at the remote location to be acceptable the trip will be approved. However, security does not stop there; rather, the visiting dignitary will also bring his or her own security detail when traveling. This security team will provide an additional security layer, in excess of local security safeguards, around the foreign dignitary while he or she is in the country.

PCPP performs two similar steps. First, the prospective host computer is scanned to measure its security capabilities and the associated risk of executing on the host. Applications are only launched on the host if the associated risks are deemed acceptable. Second, before, during, and after execution on the host computer PCPP actively denies third parties' access to all PCPP application space, i.e. PCPP owned memory areas.

The body of this article further defines and describes PCPP. We start with a related work section in which three public platforms, grid computing, mobile agent computing, and USB flash key chain portable applications, are reviewed. We then discuss risk versus trust. Next we define PCPP by describing the PCPP remote host assessment and the PCPP active security properties. Finally, we offer a section on future work and conclusions.

II. Related Work

A. Grid Computing

Grid computing allows public and private entities to pool computing resources and share these resources among large user communities across a wide geographic area. Globus [1] offers a popular toolkit which can be used to create a working grid.

The Grid Security Infrastructure (GSI) [3] is a collection of specifications, architecture, and implementations which

define available security for Globus grids. Globus grids are access controlled environments which use PKI for authentication and TLS for access control and message confidentiality. When TLS is not available gateways are used to translate TLS for use with other security architectures. An example of this is Kerberos credential translation to and from the GSI TLS/PKI framework.

A Globus grid user is typically sponsored by another individual and must obtain a PKI certificate from a certificate authority [CA]. The servers available on the Globus grid are also tightly controlled. Servers on grids are added in blocks, usually in large scale sharing arrangements between large entities such as universities or corporations.

B. Mobile Agent Security

Mobile agents are self contained, often platform independent programs which traverse a network working autonomously towards a single objective. Because of the differences between mobile agents and ordinary stationary computer programs mobile agents have a unique set of security challenges [13].

Mobile agents are difficult to trace to an individual owner or developer which makes authentication and access control difficult [13]. Agents may be signed by their owner to enable authentication. Since mobile agents can carry their execution state from host to host, they may be signed by every host on which they execute. Since authenticating this string of signatures is burdensome researchers have proposed alternative methods to assist an execution host in gaining trust of an agent. Necula et al. propose using proof carrying code which allows the agent to formally prove that it adheres to a set of security properties [17]. Sandboxing the agent is another way the host can minimize risk from the agent. With sandboxing the agent can be constrained to operate within a predefined safe zone on the host.

Mobile agents may not trust the execution host [13]. While executing on a foreign host the mobile agent is essentially under the control of that host. Bierman et al. [15] classify threats from malicious hosts into four categories. Hosts may alter the agent's control flow and or data, hosts may deny or delay service to the agent, hosts may eavesdrop and or record agent control flow or data to steal agent secrets or reverse engineer the application, and hosts may masquerade as a different host or attempt to steal the identity of the agent and clone it for its own purposes

There have been many approaches proposed for solving the above set of problems. In [13] Chess calls for a system of reputation servers to vouch for the remote servers. In [16] Wang et al. offer code obfuscation as a means to block reverse engineering of the mobile agent code. In [9] Sander et al. propose a method for encrypting mobile code in a manner such that the resulting cipher can be executed as is and return a ciphered result. State appraisal [17], cryptographic traces [18], and holographic proofs [19] have been proposed to prove post execution that an agent executed correctly and was not altered during execution. These approaches all require significant amounts of data be returned to the owner of the agent to evaluate execution correctness. A fault tolerant approach replicates the agents [20] and then uses voting to determine the correct response.

C. USB Key Chain Portable Applications

Software wrappers have been developed which allow applications to run directly from mounted USB flash drives [2] [4]. A key selling point for these wrappers is privacy. The solutions avoid leaving application data behind on the host computer by not writing data on the host when possible and requiring the application wrapper to clean up any temporary files created on the host during execution.

These wrappers take a step in the right direction for privacy. However, none of these approaches stop malicious host or applications running on the host from altering, monitoring, or recording the application's control flow and or data.

D. Trust vs. Risk

The terms trust and risk are widely used in the area of computer security. In fact approaches to solving the computer security problem can be generally divided into two categories, those which achieve security through trust, and those which achieve security through risk mitigation.

Grandison et al. define trust as "the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context" [9]. A key idea in this definition is that trust is not active, trust is a belief. For instance, if a system has complete trust in a user, then the system believes that the user will always act dependably, securely, and reliably within the context of the system. One problem with strictly trust based systems is, since a user or systems next action can not be predicted, it is not possible to have complete trust in any user or system. Since, trust is not active there is no mechanism in place in a purely trust based system to stop a user or system from acting in a malicious manner. Because of this risk mitigation is necessary.

Risk is defined in the Oxford English dictionary as "the chance or hazard of commercial loss" [12]. Risk mitigation is the practice of actively attempting to reduce risk. Risk mitigation identifies potential hazards and actively monitors conditions related to those hazards to ensure the hazard does not become a loss.

We classify PKI [10] and PGP [11] as trust based methodologies. Alternatively, we see virus protection, firewalls, intrusion detection systems, fault tolerance, and encryption as risk mitigation based methodologies.

In, PKI [10], a certificate authority issues certificates to the communicating parties which strongly identify the end user/entity. Here, we use strongly to note that a certificate authority takes great pains to validate the identity of the end user/entity. Knowledge of the user/entity's identity

encourages the end user/entity to act within community norms and not violate the trust established between the two parties. The end user/entity knows its actions can be tracked back to its real identity and therefore online and offline repercussions can come of its actions.

PGP uses a reputation based trust framework [11]. The reputation based approach requires only knowledge of a user's online identity. Here multiple third party opinions of the individual are used to build a user's reputation. The user gains privilege over time by continually adhering to the set of norms important to the community of peers. Such behavior contributes to his reputation, while violating the community norms degrades his reputation.

Both the strong identity and the reputation approach build trust on social engineering principles. The basic premise is that if you know a person, or your friends vouch for a person, then that person can be trusted. This social engineering model is often foiled in personal relationships and has similar flaws in computational environments.

The alternative to achieving security based on a trust network is achieving security through risk mitigation. Virus protection, firewalls, intrusion detection systems, fault tolerance, and encryption all attempt to actively reduce risk. Virus protection both scans files for virus signatures and monitors computer activities watching for warning signs of virus behavior. Firewalls actively close certain ports on a computer to stop certain traffic, and actively monitor packets transmitted through the firewall looking for certain attack signatures. Intrusion detection systems actively monitor a system scanning for signs of an intrusion. If an intrusion is detected various actions can be triggered. Fault tolerance systems handle faults as they occur, in real time. This can be done in multiple ways, but one common approach is through redundancy. Finally, encryption enables privacy by allowing only entities in possession of an encryption key to read a message.

III. Private Computing on a Public Platform

Private computing on a public platform (PCPP) allows users to run applications on machines previously unknown to the PCPP user. PCPP does this by first assessing the potential host to understand potential risks and capabilities associated with the host. PCPP uses historical data from prior PCPP runs on a number of separate hosts to predict whether the new host is a threat or non-threat. PCPP then brings its security along during execution, by actively enforcing a set of properties which allow PCPP to assert that both the control flow and data flow of its application have remained unaltered, unmonitored, and unrecorded for future analysis.

In this section we define the term private computing on a public platform (PCPP) by dissecting it into two parts, private computing and public platform. Next we continue defining PCPP via discussions on the remote host security assessment and the active security properties.

A. Private Computing

The term privacy varies according to the context in which it is used. Anderson et al. [5] define privacy as the ability and/or right to protect your personal secrets. Further, Anderson extends the definition of privacy, adding the ability and/or right to prevent invasions of your personal space. PCPP provides the ability to protect privacy by adhering to both parts of Anderson's definition of privacy, protecting your personal secrets and preventing invasions of your personal space.

PCPP protects personal secrets by keeping all PCPP data and meta-data confidential from third parties. Here, third parties are defined not only as eavesdroppers on the communication link between the client and host computers, but also current and future users of the host computer.

B. Public Platform

We define a public platform as a system or host which offers services to a foreign user. Neither the public platform software nor hardware configuration is controlled by the remote user.

Hosts in grid computing networks, hosts which run mobile agents, and hosts which run USB key chain applications are all public platforms.

C. PCPP Remote Host Assessment

Before PCPP will launch an application on a remote host it must first evaluate the proposed host's available security safeguards and judge the relative risk of using the host. PCPP does this via a two step process. First, PCPP confirms a set of go/no-go properties about the host. These properties are a minimal set of system capabilities and security requirements. Second, PCPP classifies the prospective host as a threat or non-threat based upon a larger set of data collected from the remote host and compared to historical PCPP data known to the PCPP client.

The security requirements from the go/no-go scan include requiring certain OS, up to date OS patches, specific virus scanners and specific virus scan history, among other security items. The prospective must not be a host known to have acted maliciously in the past. The go/no-go requirements listed here are not complete. These will be further defined in future work. The go/no-go requirements are also not static.

The second portion of the remote assessment uses a larger set of attributes collected from the remote host to predict whether the remote host is a threat or non-threat. This classification is performed with a Naïve Bayesian classifier which uses as training data a database containing the same attributes from previous PCPP runs. The historical data also contains classification of the previous runs as either threat or non-threat. This historical classification is based upon experience while running the PCPP application on the remote

host, and is specifically derived from whether or not the PCPP active security properties detected a threat on the remote host.

D. PCPP Active Security Properties

Similar to the foreign dignitary from the introduction, PCPP carries with it, its own security detail. By ensuring that the PCPP active security properties are enforced during PCPP application execution, privacy is guaranteed.

The PCPP active security properties are broken into two groups. First, a set of properties are designed to ensure that the PCPP application's control flow runs without being monitored or altered by any application running on the host. The second set of properties ensures that all application data is not accessed, altered, or recorded by any non-PCPP application running on the host. The monitoring and enforcement of the PCPP active security properties precedes application execution, and endures through out the life of the application.

The PCPP active security properties apply to the user application running under PCPP protection and to all secondary PCPP applications and monitors themselves.

TABLE 1: PCPP CONTROL FLOW PROPERTIES

	PROPERTY
1	Order of instruction execution must not be monitored or altered
2	Memory accesses made by PCPP must not be monitored or altered
3	PCPP applications may not be paused or slowed for significant amounts of time
4	Emergency shutdown/ clean-up upon control flow violation

A PCPP application's control flow should not be monitored or altered. Monitoring of the control flow may allow a third party to glean private information about the application. This can lead to knowledge of what the application is or what it does. Altering the control flow may result in destroying the integrity of any data returned by the PCPP application. It may also allow a third party to circumvent PCPP security and allow unauthorized access to sensitive information about the application or data.

To monitor and protect the control flow we must ensure three things. First, the order in which instructions are executed by the PCPP application must not be monitored or altered. Second, memory accesses made by the PCPP application may not be monitored or altered. Third, PCPP applications should not be paused for significant amounts of time.

If a PCPP application is paused for a significant amount of time this could be an indication that a third party is attempting to circumvent PCPP security. All PCPP applications will monitor the paused time of other PCPP applications. An interdependence scheme will be developed such that all PCPP applications are monitored by at least one other PCPP application. A PCPP application assigned to monitor a second PCPP application will expect periodic communications from the PCPP application being monitored.

The periodic communication requirement enables detection of a killed PCPP application, a slowed PCPP application, and a paused PCPP application due to single stepping and break pointing. If a PCPP application is killed, slowed, or paused the PCPP emergency shutdown procedure is called.

The maximum amount of time that an application can be paused must be set to allow for multithreading.

When a control flow violation is detected PCPP immediately executes an emergency shutdown procedure. This emergency shutdown procedure includes stopping all PCPP applications and erasing all data on the host. During emergency shutdown data is erased according to the following priority. First, all encryption keys will be destroyed. Second, all unencrypted data will be destroyed. Third, all encrypted data will be destroyed. Fourth, all communication links will be broken. Finally, the PCPP client will shutdown.

TABLE 2: PCPP DATA PROTECTION PROPERTIES

	PROPERTY
1	Encryption of all possible input and output channels to and from a PCPP application
2	Encryption key protection
3	Register all non-encrypted channels
4	Destroy all data stored on host after application execution
5	Detect unauthorized memory access
6	Emergency shutdown/ clean-up upon threat detection

The PCPP data protection properties listed in Table 2 are described in the following section.

Whenever possible data into and out of the PCPP application must be encrypted. This encryption is intended to make it difficult for a rogue to make sense of any data which was captured despite other defenses in place by PCPP. Typical channels which will require encryption include data stored in temporary files on the host machine and communication streams into and out of the host, such as internet connections. PCPP applications may also encrypt data stored on an application stack and data stored in the heap.

Since, much of the data into and out of a PCPP application is to be encrypted PCPP must secure the encryption keys used for this process. Details on how the encryption keys a protected is left for future work.

All non-encrypted input and output channels must be registered with the PCPP client. This registration process ensures that system monitors watch these channels for threats and makes destruction of data related to these channels a priority during PCPP shutdown.

After PCPP application execution all PCPP data must be destroyed. This includes the destruction of all input and output from the PCPP application, all temporary files created

during execution, and all application code downloaded to the host machine prior to execution. The term destruction is used rather than erase to acknowledge that merely placing files in a host's recycle bin is not enough. Files should be thoroughly erased. This means overwriting memory areas used by PCPP applications, overwriting the frame buffer, and overwriting any files written to disk drives.

A mechanism must be in place to know if areas of memory which store PCPP application information are being accessed. These monitor functions must be able to inform the PCPP client of a threat in real time to allow the PCPP client to respond to the threat.

If a threat is detected by PCPP system monitors the PCPP client execute the emergency shutdown procedure mentioned earlier.

The monitors and controls used by PCPP to ensure the proper control flow and to protect PCPP data, effectively create a container around the PCPP application. This virtual environment creates a partition between the host and the PCPP application. Besides monitors and controls the PCPP container maps physical addresses of system peripherals to logical addresses used by the PCPP application. This logical address scheme provides a consistent environment which enables portability of the PCPP application. A second benefit of the PCPP container is provided to the host; by containing the host in a preset virtual environment the PCPP application's ability to access and alter sensitive areas of the host is minimized.

The PCPP active security properties work to eliminate the risk of privacy invasion by the host machine. These properties do not combine to completely eliminate risk. Rather, the properties decrease risk, and where directly decreasing risk is not possible the PCPP active security properties attempt to inform the PCPP client of risks as they occur so that an application may be flushed. The PCPP client will dump all applications and commence with a system clean-up immediately upon learning of a privacy threat.

IV. Future Work

Next we plan to further define the remote security assessment including details on the go/no-go properties and the Naïve Bayesian classifier used to classify remote hosts as a threat or non-threat.

Research is underway to examine how one will enforce the three primary properties of PCPP active security i.e., the application data and control flow must run unaltered, unmonitored, and unrecorded on the remote host.

Finally, we plan an implementation of PCPP and an evaluation of its threat avoidance effectiveness, its impact on the execution time of the PCPP applications, and its potential to reduce an entities' directly owned compute resource requirement.

V. Conclusion

We have introduced private computing on a public platform (PCPP), a new paradigm in portable application security. In this paper, we defined PCPP as a two step process which ensures application privacy on a remote host which is out of the control of the PCPP user. First, PCPP reviews the remote host security and system configurations to determine if the remote host is an acceptable risk. While running on a remote host PCPP enforces a set of active security properties. These properties ensure that the PCPP application data and control flow remain unaltered, unmonitored, and unrecorded.

We have shown that PCPP allows a user to run applications on a remote host which was previously unknown to the user, thereby increasing the number of potential execution hosts available to a portable application and making PCPP a good fit in an era of increasing portable computing.

References

[1] www.globus.org
[2] www.u3.com
[3] Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K., Gawor, J., Kesselman, C., Meder, S., Pearlman, L., and Tuecke, S. Security for Grid Services. In International Symposium High Performance Distributed Computing (Seattle, WA, June 2003), pp. 48--57.
[4] http://www.portableapps.com
[5] Anderson, R. J. 2001 Security Engineering: a Guide to Building Dependable Distributed Systems. 1st. John Wiley & Sons, Inc.
[6] Wang, C. and Wulf, W. A, "A Framework for Security Measurement." Proc. National Information Systems Security Conference, Baltimore, MD, pp. 522-533, Oct. 1997.
[7] T. Sander, C. Tschudin, Protecting mobile agents against malicious hosts. In G. Vigna (ed.) Mobile Agents and Security, LNCS, to appear.
[8] Chess D., B. Grosof, C. Harrison, D. Levine, C. Parris and G. Tsudik, Itinerant Agents for Mobile Computing. Technical Report, October 1995, IBM T.J. Watson Research Center, NY.
[9] T. Grandison and M.Sloman, A survey of trust in Internet application, IEEE Communications Surveys, Fourth Quarter, 2000.
[10] Adams, C. and Lloyd, S. 1999 Understanding Public-Key Infrastructure: Concepts, Standards, and Deployment Considerations. Macmillan Technical Publishing.
[11] Zimmermann, P. R. 1995 The Official PGP User's Guide. MIT Press.
[12] Oxford English Dictionary, http://dictionary.oed.com/
[13] Chess, D. M. 1998. Security Issues in Mobile Code Systems. In Mobile Agents and Security G. Vigna, Ed. Lecture Notes In Computer Science, vol. 1419. Springer-Verlag, London, 1-14.
[14] George C. Necula, Peter Lee, Safe, Untrusted Agents Using Proof-Carrying Code, Lecture Notes in Computer Science, Volume 1419, Jan 1998, Page 61
[15] Bierman, E. and Cloete, E. 2002. Classification of malicious host threats in mobile agent computing. In Proceedings of the 2002 Annual Research Conference of the South African institute of Computer Scientists and information Technologists on Enablement Through Technology (Port Elizabeth, South Africa, September 16 - 18, 2002). ACM International Conference Proceeding Series, vol. 30. South African Institute for Computer Scientists and Information Technologists, 141-148.
[16] Wang, C., Hill, J., Knight, J., and Davidson, J. 2000 Software Tamper Resistance: Obstructing Static Analysis of Programs. Technical Report. UMI Order Number: CS-2000-12., University of Virginia.
[17] Farmer, W. M., Guttman, J. D., Swarup, V. 1996b. Security for mobile agents: Authentication and state appraisal. In Proceedings of the Fourth

European Symposium on Research in Computer Security (ESORICS), E. Bertino, H. Kurth, G. Martella, and E. Montolivo, Eds., Lecture Notes in Computer Science, vol. 1146, Springer-Verlag, New York, 118–130.

[18] Vigna, G. 1997. Protecting mobile agents through tracing. In Proceedings of the Third ECOOP Workshop on Mobile Object Systems: Operating System Support for Mobile Object Systems.

[19] Yee, B. S. 1999. A sanctuary for mobile agents. In Secure Internet Programming: Security Issues for Mobile and Distributed Objects, J. Vitek and C. Jensen, Eds., Lecture Notes in Computer Science, vol. 1603, Springer-Verlag, New York, 261–274.

[20] Minsky, Y., Van Renesse, R., Schneider, F. B., and Stoller, S. D. 1996. Cryptographic support for fault-tolerant distributed computing. In Proceedings of the Seventh ACM SIGOPS European Workshop, 109–114.

[21] Morris, T., Nair, V.S.S., Private Computing on a Public Platform, Department of Computer Science and Engineering, Southern Methodist University, Technical Report 06-CSE-01, 2006