Proceedings of the 2003 IEEE
Workshop on Information Assurance
United States Military Academy, West Point, NY June 2003

The TrustedFlow™ Protocol
Idiosyncratic Signatures for Authenticated Execution

Mario Baldi[°], *Member, IEEE*, Yoram Ofek[◊], *Member, IEEE*, and Moti Yung[□]

Abstract – This work presents a software solution to the problem of remotely authenticating software during execution, which aims at assuring that the software is not changed prior to and during execution. The solution is based on a flow of idiosyncratic signatures that is generated by a function hidden in the software to be authenticated and validated by a remote computing component. The TrustedFlow™ approach is complementary to many current enhancements for secure computing and networking: while other approaches provide privacy and authentication protecting from the attacks of a man in the middle**, TrustedFlow™ protects from the attack of a** man at the edge**.**

Index terms – trusted code execution; total security for networked applications; trusted computing

I. Introduction

Software, especially in the context of data networks, suffers from some inherent problems. These include modifications, either by a malicious or inadvertent attacker, malware distribution (e.g., viruses and Trojan horses), and the use of malicious software remotely for penetration, intrusion, denial-of-service (DoS), and distributed DoS (DDoS).
TrustedFlow™ is a software solution to the problem of remotely authenticating code of software procedures and protocols during execution, which aims at assuring that the software is not changed prior to and during execution. The solution is achieved by continuously emanating a flow of idiosyncratic signatures that authenticate the software from which they have emanated. The idiosyncratic signatures are generated by a secret function that is hidden (e.g., obfuscated) in the software and whose execution is subordinated to the proper execution of the software being authenticated. The flow of signatures is validated at a remote computing component. This generation and validation method of idiosyncratic signature is called *TrustedFlow™ protocol*. The TrustedFlow™ protocol is a general add-on protection tool that complements other security tools such as trusted computing platforms, authentication and encryption protocols. No other authentication method (to the best of our knowledge) certifies the software continuously during run-time by emanating idiosyncretic signatures. In other words, while other approaches provide privacy and authentication protecting from the attacks of a man in the middle, TrustedFlow™ protects from the attack of *a man at the edge*. The TrustedFlow protocol has broad *synergistic implications* on various computing and networking protection means.
The TrustedFlow™ protocol has broad potential applications in grid-computation, intrusion avoidance, digital right management, and the protection of server applications from misbehaving client programs (e.g., malware SSL and IPSec). Possible applications include (1) DoS and intrusion avoidance assuring trusted clients, (2) VPN add-on assuring correct software execution (malware free), (3) Protecting client applications (from clients) in content handling programs (digital right management - DRM), (4) Protecting server applications from misbehaving client programs, such as, malware SSL, malware IPSec, (5) Protecting (Java) applets in Peer-2-Peer collaborative computing, (6) Trusted network management, and (7) Protecting the routing infrastructure — avoiding misuse of the various routing protocols — and (8) Trusted traffic conditioning, i.e., remote verification of compliance to a traffic profile.

II. TrustedFlow™ Protocol Principles

This solution has two basic components. The first is the preparation and manipulation of the software code at the source. This is needed to assure that the idiosyncratic signature generation is bound in an inseparable manner to the actual functional code. The first component generates a "*combined module*" to be executed at run time. The second component is the signature generation and checking during run-time. The two above components are based on the following.

- **Interlocking** is the general term we use to describe a combining of original executable software modules (original function) together with an idiosyncratic (pseudo-random) signature generator (which is used to generate an *unpredictable flow of tags*) forming a "*combined module*."

[°] *Computer Engineering Department, Torino Polytechnic, Italy*
[◊] *Synchrodyne Networks, Inc. New York, NY*
[□] *Computer Science Department, Columbia University, New York, NY*

ISBN 555555555/$10.00 © 2002 IEEE

- **Program Hiding** the *"combined module"* is prepared in such a way as to ensure that reverse engineering is difficult enough so that it becomes functionally infeasible.

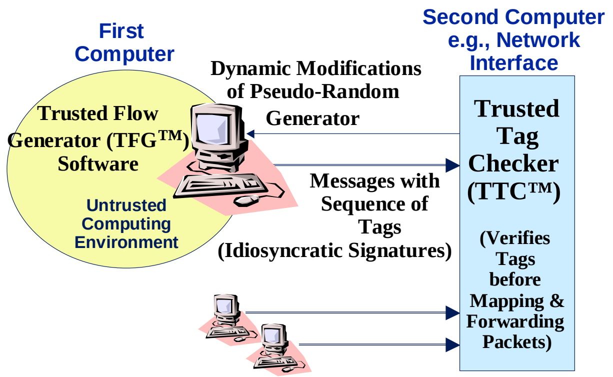
First Computer
Trusted Flow Generator (TFG™) Software
Untrusted Computing Environment
Dynamic Modifications of Pseudo-Random Generator
Messages with Sequence of Tags (Idiosyncratic Signatures)
Second Computer e.g., Network Interface
Trusted Tag Checker (TTC™)
(Verifies Tags before Mapping & Forwarding Packets)

Figure 1: TrustedFlow™ networking and computing

During run-time, the following two components define the TrustedFlow protocol:

- A **Trusted Flow Generator** (**TFG**) (see Figure 1 and Figure 2) executes the "*combined module*", which constantly outputs its results (such as messages) together with a flow of tags, constituting the continuous idiosyncratic signature of the *"combined module's"* run-time execution.
- A **Trusted Tag Checker** (**TTC**) (see Figure 1 and Figure 2) is used to remotely authenticate (verify) the flow of idiosyncratic signatures that forms the continuous idiosyncratic signature emanated from the "*combined module*," thus assuring that the correct "*combined module*" was executed in run-time.

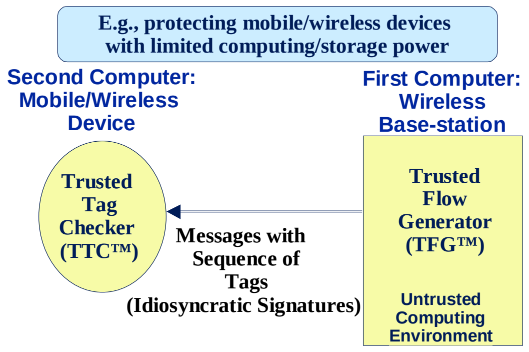
E.g., protecting mobile/wireless devices with limited computing/storage power
Second Computer: Mobile/Wireless Device
Trusted Tag Checker (TTC™)
First Computer: Wireless Base-station
Trusted Flow Generator (TFG™)
Untrusted Computing Environment
Messages with Sequence of Tags (Idiosyncratic Signatures)

Figure 2: "Reverse" TrustedFlow — protecting mobile devices

A "**reverse**" TrustedFlow protocol, as shown in Figure 2, is a variant of the TrustedFlow™ protocol, such that, the TTC function is not in the network interface but in the end device. The "**reverse**" TrustedFlow can be used for protecting mobile/wireless devices. Such devices typically feature limited computing and storage capabilities. In this case, the mobile device contains the TTC functionality in order to off-load in a trusted manner some computations, such as anti-virus programs, to a wireless base-station, which contains the TFG functionality. The rationale is that the base station has vast computing resources, while mobile devices are limited in computing and storage capabilities.

As mentioned, the TFG contains a hidden program that is obfuscated into the program used for generating and sending messages. More specifically, the TFG contains the program for sending messages, which contains an obfuscated secret part that generates a pseudo-random sequence of n-bit tags (idiosyncratic signature). Only the TTC is able to locally generate the sequence of n-bit tags. The TTC then compares the locally generated sequence with the received sequence. A successful comparison verifies that the legitimate program was used to generate and send the messages. Next, we briefly discuss what is obfuscation, which is one of the key components of the TrustedFlow protocol.

III. Creating a Trusted System Environment

TrustedFlow™ (hence the TFG and TTC) is a general software tool with wide commercial and military applications. Through a novel use of idiosyncratic signatures the TrustedFlow™ protocol can ensure:

(i) That users cannot change the software modules that compute, generate, and send messages; and

(ii) That users cannot abuse the way various protocols (such as, E-MAIL, PING, HTTP, TCP) are intended to be used for generating messages and sending messages – thereby avoiding, for example, the DoS/DDoS/intrusion phenomena.

In essence, the TrustedFlow™ protocol creates a combined trusted computing/networking environment by using an idiosyncratic signature that is checked by the TTC component operating in a trusted environment (e.g., firewall) together with an idiosyncratic signature, which is generated by a TFG component operating in an untrusted environment. TrustedFlow™ complements trusted computing platforms by enabling remote verification that the right software is executed, and in effect is constructing a "remote trust" mechanism. The TrustedFlow™ protocol has low complexity (no need for special hardware), and thus, it is less expensive and more scalable than other solutions.