Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Trustworthy clients: Extending TNC to web-based environments

Sascha Rehbock, Ray Hunt *

University of Canterbury, Computer Science and Software Engineering, Private Bag 4800, Christchurch, New Zealand

ARTICLE INFO

Article history:
Received 8 May 2008
Received in revised form 22 December 2008
Accepted 24 December 2008
Available online 15 January 2009

Keywords:
TNC
Integrity measurement
Web applications
SAML
WS-Trust

ABSTRACT

In Trusted Network Connect (TNC), a network access decision is based on the security state of an access requesting party. This mechanism is limited to closed environments such as LANs and VPNs. In this paper, we propose solutions based on authentication standards for enabling TNC in open, web-based scenarios. In particular, an architectural model for TNC is proposed that takes additional security and privacy requirements into account. Furthermore, a communication scheme is proposed that is based on standardised protocols and message formats. This approach provides assurance as to the security state of clients accessing security sensitive web-based services.

1. Introduction

Trusted Network Connect (TNC) [19] is an emerging technology for Network Access Control (NAC). Traditional NAC systems focus mainly on authentication mechanisms, that is, on preventing unauthorised devices connecting to a local area network. However, they fail to prevent the possibility of network devices with a questionable state of security connecting to the network. Network devices that are not in a current security state, i.e. without security updates, proper anti-virus measures, or personal firewall software, can threaten the security of servers and other workstations that are connected to the same network. TNC addresses this issue by making network access to a device dependent on the device's state.

However, TNC is currently limited to local area network and VPN scenarios. That is, the TNC specifications and implementations thereof only allow TNC to be used in controlled environments using network link level protocols. Web-based applications are an interesting new area for TNC because they are widely used and they often involve processing of security sensitive information. In this paper, we use the term *web-based* to refer to systems and applications in open environments that are based on Internet technology and accessed using public networks. Examples include Internet banking, e-government applications, and remote access to corporate networks.

Malicious software installed on a PC can bypass other security mechanisms (such as transport layer encryption) and alter application requests. It is hence capable of altering online banking transactions and accessing confidential information in an Extranet on behalf of the user [15].

An architecture that supports the use of TNC in a web environment must be based on open standards in order to allow easy integration and thus gain acceptance. Taking TNC in LAN environments as an example, this means that a web-based TNC solution has to be integrated into existing authentication mechanisms. The TNC integrity check can be performed after an initial authentication. The result of the integrity check (e.g. that the PC is in a secure state) can then be treated as an attribute of the authentication process. This integrity check does not replace the need for existing authentication processes or obviate the requirement for overall web-service security in general.

The aim of TNC is to base a network access decision not only on authentication but also on the security state of an endpoint. In the context of TNC, this security state is called *integrity*. TNC defines integrity as the "relative purity from software that is considered harmful". Anti-malware software, personal firewalls, and security updates can protect an endpoint. Their presence can thus be used as an indicator that an endpoint is not infected with malicious software [19].

TNC is designed as an open standard to allow interoperability between different vendors. This openness allows TNC to be extended and used outside of local area network environments. As an open standard, TNC has the potential to become a ubiquitous system for endpoint integrity measurement. Such a system has the potential to shield systems and their users from the effects of malicious software.

In this paper, we propose a protocol stack that enables the use of TNC in web-based environments. Furthermore, we propose changes to the TNC architecture to reflect additional security and

* Corresponding author. Tel.: +64 3 3642347.
E-mail addresses: sur21@student.canterbury.ac.nz (S. Rehbock), Ray.Hunt@canterbury.ac.nz (R. Hunt).

0140-3664/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2008.12.040

privacy requirements. Overall, such a system can help to mitigate security risks arising from malicious software, and can protect users in both corporate and non-corporate environments.

After a brief introduction to TNC in the next section, various architectural models are compared in Section 3. Based on this architecture, a communication scheme for performing a TNC integrity check is proposed in Section 4.

Sections 5 and 6 describe the proposed set of protocols and message formats that implement the scheme. Finally, Section 7 summarises the paper and gives an overview of further steps in this ongoing research project.

2. TNC overview

The architecture of TNC has been designed to be vendor and technology neutral. The architecture, as specified in [19], incorporates the following three entities.

The **Access Requestor (AR)** is a client, for example a PC or a notebook, seeking access to the network. Its integrity state is measured and reported to the Policy Decision Point. The **Policy Decision Point (PDP)** is the entity which decides whether the AR should be granted access to the network. These decisions are based on policies which describe the integrity requirements that a client must fulfil. A **Policy Enforcement Point (PEP)** is the entity, for example a switch, responsible for enforcing the decision made by the PDP.

Three phases exist in the TNC architecture: *Assessment, Isolation* and *Remediation*. During these phases the TNC entities and their components interact with each other as depicted in Fig. 1 (based on [19, Fig. 2]). This interaction is next briefly described.

The assessment phase is started by an AR wishing to gain access to a protected resource (e.g. a network). In this phase, integrity measurements are gathered on the AR by *Integrity Measurement Collectors (IMCs)* and accumulated by the *TNC Client (TNCC)*. The TNCC encapsulates the measurements in an XML-based format and sends them to the *TNC Server (TNCS)* on the PDP. These XML-based messages, exchanged between a TNCC and a TNCS, are called *TNCCS-Batches* as defined in [22, Section 2.8 seqq.].

Every IMC on the AR has a counterpart on the PDP, called an *Integrity Measurement Verifier (IMV)*. The TNCS delivers measurements to the related IMV, which compares a measurement to the network access policy. This policy states the minimum security requirements that must be satisfied by a client wishing to access the network. Before an IMV is able to compare the security state of an AR with a policy, it might be necessary to request further measurements from an IMC, thus requiring several round-trips.

The results of all comparisons are combined by the TNCS and sent to the *Network Access Authority (NAA)*. The final access decision is made by the NAA. This decision is sent to the *Policy Enforcement Point (PEP)*, which is then responsible for enforcing it.

If the integrity of an AR does not comply with the policy, the isolation phase can be initiated. In this case, the PEP is instructed by the NAA to restrict client access to a certain isolated part of the network. After an AR has been isolated, the remediation phase can commence. In this phase, integrity-related updates are sent to the AR. As soon as these updates have been installed successfully, the assessment phase is re-initiated, and the AR can be granted access to the network.

3. Adapting the architecture

The original TNC model, as described in the previous section, cannot be used directly for web-based use cases. This is because there is (a) no centralised Policy Enforcement Point (e.g. a switch) that handles all access requests, and (b) rather than accessing one entity (i.e. one network), a user will typically access multiple services (e.g. online banking, e-commerce, and e-government sites). Because of this different environment, the following extended requirements for a TNC-based architecture apply.

Trustworthy Verifier. A TNC integrity check reveals detailed information about the software installed and executed on a PC. A malicious verifier could use this information to perform an attack by targeting known vulnerabilities of these software products. Knowing, for instance, that a system has not had the latest operating system patches applied, may give an attacker opportunities to attack. In a web-based scenario, potentially any service offering party (including malicious parties) can request an integrity check. It is therefore important to protect the user from the effects of performing an integrity check with a malicious verifier.

Privacy concerns. As stated above, detailed information about a PC is revealed during an integrity check. In an uncontrolled network, such as the Internet, a user typically interacts with a number of parties, and may not want to disclose integrity information to them. It is hence important to integrate privacy protecting mechanisms into any scheme.

Usability and user experience. Unlike in a corporate environment, IT service staff are not available for users of Internet-based services. It is therefore essential to keep the process of TNC integrity checking simple and transparent for the user.

The original TNC architecture has to be adapted to fulfil the above requirements. While it might not be possible to completely satisfy every requirement, workable compromises have to be

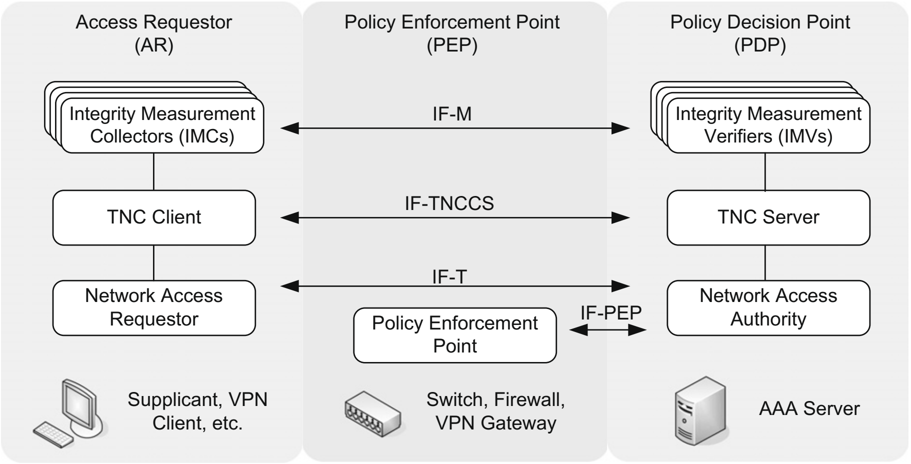
Access Requestor (AR)
Policy Enforcement Point (PEP)
Policy Decision Point (PDP)
Integrity Measurement Collectors (IMCs)
IF-M
Integrity Measurement Verifiers (IMVs)
TNC Client
IF-TNCCS
TNC Server
Network Access Requestor
IF-T
Network Access Authority
Policy Enforcement Point
IF-PEP
Supplicant, VPN Client, etc.
Switch, Firewall, VPN Gateway
AAA Server

Fig. 1. Overview of the TNC architecture.