Trustworthy Clients: Architectural Approaches for Extending TNC to Web-Based Environments

Sascha Rehbock
Computer Science and Software Engineering
University of Canterbury
Private Bag 4800, Christchurch, New Zealand
sur21@student.canterbury.ac.nz

Ray Hunt
Computer Science and Software Engineering
University of Canterbury
Private Bag 4800, Christchurch, New Zealand
Ray.Hunt@canterbury.ac.nz

ABSTRACT
Trusted Network Connect (TNC) is a network access control mechanism that takes the security state of access requesting endpoints into account. This mechanism is currently limited to local area networks and VPN connections. However, TNC can further be useful in other scenarios, such as security sensitive web-based applications. An architectural approach for TNC in this environment based on authentication standards is presented in this paper.

Categories and Subject Descriptors
D.4.6 [Operating Systems]: Security and Privacy Protection—authentication, verification

Keywords
TNC, Integrity Measurement, Web Applications, SAML

1. INTRODUCTION
Trusted Network Connect (TNC) is a recently emerging technology for Network Access Control (NAC). Traditional NAC systems focus mainly on authentication mechanisms, that is, preventing unauthorised devices to connect to a local area network. However, they fail to prevent that network devices with a questionable state of security can be connected to the network. Network devices that are not in a proper security state, i.e. without security updates, proper anti-virus, and personal firewall software, can threaten the security of servers and other workstations that are connected to the same network.

The aim of TNC is to base a network access decision not only on authentication but also on the security state of a network 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.[9]

In contrast to manufacturer-specific solutions1 TNC was designed as an open standard to allow interoperability between different vendors. What is more, 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 caused by malicious software.

A step towards such a system is to broaden the scope of TNC. Currently, TNC is limited to local area network and VPN scenarios. Web-based applications are an interesting area for TNC because they are widely used and many of them involve processing of security sensitive information. 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. 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 to 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 can then be treated as an attribute of the authentication process (e.g. PC is in a secure state).

In this paper an architecture is proposed that allows the integration of TNC in web scenarios. The advantages of TNC in local area networks can thus be used in a broader context. Overall, such a system can help to mitigate security risks arising through malicious software, and can protect users in corporate and non-corporate environments. After a brief introduction to TNC in the next section, different architectural models are compared in section 2. A suitable implementation mechanism is proposed in section 3. Finally, section 4 summarises this paper and and gives an overview of the further steps in this ongoing research project.

1.1 TNC Overview
The architecture of TNC has been designed as a framework in order to be vendor and technology neutral. The architecture, as specified in [9], is composed of the following three

---

1such as Cisco’s Network Admission Control (NAC) and Microsoft’s Network Access Protection (NAP).
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 makes the decision about whether the AR should be granted access to the network. These decisions are based on policies which describe the integrity requirements a client must fulfill. A Policy Enforcement Point (PEP) is the entity, for example a switch, that is responsible for enforcing the decision made by the PDP.

Three phases exist in the TNC architecture: Assessment, Isolation and Remediation. They are briefly described in the following.

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 and are sent to a PDP. Each measurement is compared to the network access policy. The results of the comparison are combined and an access decision is made by the PDP. This decision is sent to the 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 that case, the PEP is instructed by the PDP to restrict access to a certain isolated part of the network. After an AR is 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.

2. ARCHITECTURE

The original TNC model as described in the previous section cannot directly be used 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) there are several services (e.g. online banking, e-commerce, and e-government sites) that a user accesses. Because of this different environment, extended requirements for a TNC-based architecture exist:

Trustworthy Verifier A TNC integrity check reveals detailed information about which software is installed and executed on a PC. A malicious verifier could use this information to perform an attack targeting vulnerabilities of these software products. Knowing, for instance, that a system is not issued with the latest operating system patches gives an attacker opportunities and hints about how to attack. In a web-based scenario, potentially any service offering party (including malicious ones) can request an integrity check. It is therefore important to protect the user from 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 is interacting with several parties and does not want to disclose integrity information to each of them. It is hence important to integrate privacy protecting mechanisms.

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 fulfill the above requirements. While it might not be possible to completely satisfy every requirement, workable compromises have to be found. In the following sections different models are compared.

2.1 Direct Attestation

The Direct Attestation model, depicted in figure 1, involves two parties: A client (C) and a Service Provider (SP). In addition to providing a service (e.g. an online banking service), the SP also performs the integrity check. Before granting access to its services, the SP will request a TNC integrity check.

This model may raise privacy concerns as all integrity information is sent to the SP which can link them to a user’s identity. More importantly, making educated decisions about the security state of a user’s PC makes it necessary to keep track of different versions of security related software and their updates. This is a complex task as a large number of vulnerabilities are discovered every day [1]. As it is not within the business scope of an SP to manage security vulnerabilities and updates for their users, this approach is not very practical. It is likely that this task is instead performed by a separate third party which then solely performs the role of the Verification Service Provider (VSP). This approach is described in the following.

2.2 Relayed Attestation

In the Relayed Attestation model (figure 2), a third entity exists: the Verification Service Provider (VSP). From a user’s perspective, this model is identical to the previous model. The SP acts as a transparent proxy for the client as TNC messages are relayed through the SP between VSP and client. This approach, however, creates a performance overhead and a possible bottleneck at the SP because all messages have to be decrypted and re-encrypted before they are forwarded.

In theory, the SP does not need to understand TNC messages and simply has to forward them. However, a client cannot detect or prevent an SP from eavesdropping on these messages. Hence, the SP would still be able to determine details about the client’s software configuration, which may raise privacy concerns. What is more, a malicious SP would be able to derive attack strategies if the SP were able to perform a TNC integrity check with a client. These issues can be addressed as described in the next section.

2.3 Assertion-based Attestation

The Assertion-based Attestation model, depicted in figure 3, is similar to the model described above. However, instead of relaying traffic through the SP, the VSP and client can communicate directly. The basic idea of this model is that the SP states its security requirements in form of a (high level) policy and sends this policy to the VSP. The VSP performs
the integrity check and decides whether the client is compliant to the policy. This decision is sent to the SP. As the communication between SP and VSP is relayed through the client, mechanisms must exist to protect the confidentiality and integrity of these messages.

In this model, the SP has to trust the VSP to perform the integrity checks correctly. In addition, the user has to trust the VSP for privacy purposes. As integrity measurements messages are only exchanged between VSP and client, they are not visible to the SP which reduces privacy concerns. What is more, instead of informing every SP that requires a TNC check about the details of the software configuration, only one trusted party, the VSP, gets to know details about the configuration. SPs only get a confirmation from a VSP that the client in question complies to the policy as expressed by the SP.

In general, a VSP does not require any client authentication. However, offering the integrity check service without prior authentication can lead to Denial-of-Service (DoS) attacks in which malicious users are performing fake integrity checks that consume network bandwidth and CPU time. To mitigate the risk of a DoS attack, users have to be authenticated.²

Instead of re-authenticating a user at the VSP, the SP can issue a statement for the VSP stating that a client is already authenticated. A VSP only commences an integrity check session if a user can provide such a statement from a trusted SP. Hence, this approach can protect a VSP from unauthorised usage of its services.

3. PROPOSED IMPLEMENTATION

In the previous section, the assertion-based attestation model has been identified as a suitable architecture for performing a TNC-based integrity check. Suitable protocols and mechanisms have to be found in order to turn this architecture into a working prototype. As mentioned in section 1, the TNC integrity check can be performed as part of an existing authentication mechanism. In the following, an approach is proposed that is based on existing open standards.

The most important standard used in the following is the Security Assertion Mark-up Language (SAML)[3, 4]. In summary, SAML defines an XML-based mechanism to state authentication and attribute assertions about a user. Authentication assertions can be used to make an assertion about the fact that a user was authenticated at a certain time using a certain mechanism. Attribute assertions can contain statements about certain attributes of a user, e.g. about his or her age or credit limit. In the context of TNC integrity checks, attribute assertion can be used to transfer information about the conformance of a client to a security policy. In order for a receiver of a SAML assertion to validate the authenticity of this message, assertions are digitally signed by the sender.

This approach is depicted in figure 4 and is described in the following. In step 0, a client accesses a web-based application offered by the SP using SSL/TLS secured HTTP (HTTPS). The client is authenticated by the server using, for example, a username and password mechanism. When accessing a specially protected area within the web-based application (step 1), a TNC integrity check shall be performed to reduce the risk of malicious software interfering with the online application.

The SP triggers the integrity check by including a HTML <object> tag into the response (step 2) which includes further attributes that are used to transfer the policy file and the authentication statement as described in section 2.3.

The policy can be expressed by using the format used in WS-SecurityPolicy³. For the authentication confirmation a SAML assertion can be created. This assertion contains a pseudonym for the user and the method and time of authentication. Both the policy and the SAML assertion can be encrypted for a particular VSP that is trusted to perform the integrity check correctly. In addition, the client needs to know the address of this particular VSP. This address (in form of a URL) and the VSP’s public key are sent to the client as an X.509 certificate⁴ signed by the SP. This enables the client to check the VSP’s public key and protects it from being redirected to malicious verifiers.

The browser plug-in installed on the client picks up the integrity check request (using the HTML tag as an indicator) and contacts the VSP using the details provided by the X.509 certificate in step 3. The protocol between a client and a VSP can be based on standard TNC TNCCS⁵ messages. As the TNCCS messages are XML-based, SOAP over HTTPS is a favourable choice and can be used as the transport mechanism.

The initial integrity check request contains the encrypted SAML assertion and the policy file. After decrypting these documents, the VSP validates the assertion and processes the policy file. In step 4 and 5 the TNC measurement is performed according to the TNC specifications ([10, 11]).

²[6] and [5] conclude that authentication of IP messages can help to avoid DoS attacks at the network level. This approach can be used at higher protocol levels to protect against DoS attacks at the application layer.

⁴A statement (e.g. in XML) containing the same information and signed by the SP can be used instead.
⁵TNCCS is the message format used between AR and PDP and is defined in [12].
After the VSP has determined whether the client’s security state is compliant to the policy received from the SP, a TNC access recommendation is sent to the client (step 6). This access recommendation is contained in a TNCCS message and readable for the client. What is more, the VSP creates and signs a SAML attribute assertion containing the pseudonym received in step 3 and the result of the integrity check. This assertion is encrypted for the SP. If the client is not compliant, it can now perform a remediation process and restart the integrity check. Otherwise, it will extract the SAML document from the TNCCS message and forward it to the SP using the HTTP POST method (step 7).

The SP can decrypt the SAML document and verify the signature (using the VSP’s public key) and extract the access recommendation. Based on this recommendation, it can then enforce the access decision, that is, allow or deny access to the operation requested by the client in step 1. The TNC check is now finished and further application data can be exchanged (step 8).

Further work is necessary to complete the proposed implementation strategy. Currently it is investigated if WS-Trust can be incorporated into the message exchange. This would allow to exchange SAML tokens and transfer artefacts, such as the policy file, in a standardised way.

4. CONCLUSION AND FUTURE WORK

In this paper, different architectural approaches and their properties have been discussed. As described above, the Assertion-based Attestation model provides privacy protection to clients and limits the implementation complexity for the SP at the same time. This implementation method allows TNC to be implemented as another aspect of the authentication process of web-based applications. To prove the practicability of the proposed mechanism, a performance evaluation will be performed.

Further work will also include identifying a mechanism which is capable of translating high level policies expressed by SPs to low level policies that can be understood by TNC components.

The remediation capability is a major advantage of TNC when compared to other integrity check solution, such as TCG compliant (binary)[8], property-based[7], and semantic[2] remote attestation. This mechanism allows a client to update software components so that it conforms to the policy. This mechanism, however, is not well defined within the specification and many open issues exist. Hence, further work is necessary in which the possibilities of a web-based remediation process are explored.

5. REFERENCES