

[Simulation Modelling Practice and Theory 51 (2015) 157–169](http://dx.doi.org/10.1016/j.simpat.2014.10.011)

Contents lists available at [ScienceDirect](http://www.sciencedirect.com/science/journal/1569190X)

Simulation Modelling Practice and Theory

journal homepage: [www. elsevier.com/locate/ simpat](http://www.elsevier.com/locate/simpat)



Design and analysis on trusted network equipment access authentication protocol



Yingxu Lai [a](#_bookmark0),[⇑](#_bookmark3), Yinong Chen [b](#_bookmark1), Qichen Zou [a](#_bookmark0), Zenghui Liu [c](#_bookmark2), Zhen Yang [a](#_bookmark0)

a *College of Computer Science, Beijing University of Technology, Beijing 100124, China*

b *School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe 85287, USA*

c *Automation Engineering Institute, Beijing Polytechnic, Beijing 100176, China*

a r t i c l e i n f o

*Article history:*

Received 19 August 2014

Received in revised form 9 October 2014 Accepted 26 October 2014

Available online 7 January 2015

*Keywords:*

Trusted network equipment Authentication

Security protocol BAN logic

Attack detection model

a b s t r a c t

Cloud security is a system engineering problem. A common approach to address the prob- lem is to adapt existing Trusted Network Connection (TNC) framework in the cloud envi- ronment, which can be used to assess and verify end clients’ system state. However, TNC cannot be applied to network equipment attached to the cloud computing environment directly. To allow the network devices to access the trusted network devices safely and reli- ably, we ﬁrst developed a Trusted Network Equipment Access Authentication Protocol (TNEAAP). We use the BAN logic system to prove that TNEAAP is secure and credible. We then conﬁgure the protocol in an attack detection mode to experimentally show that the protocol can withstand attacks in the real network. Experiment results show that all the nine goals that decide the protocol’s security have been achieved.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Cloud computing environment is emerging as a key platform supporting data-intensive processing and big data analysis in information technology. The advantages of convenience, economy, and high extensibility draw the attention of more and more researchers and practitioners. However, cloud computing is just like a double-edged sword, it brings us great conve- nience, and at the same time, it also carries additional problems, such as performance, reliability, trustworthiness, and secu- rity [[1,2]](#_bookmark27). With the popularity of cloud computing, the signiﬁcance of security issues is gradually increasing, and it has become an important factor to restrict cloud applications. The solution of cloud security is complex. A common approach is to adapt existing Trusted Network Connection (TNC) [[3]](#_bookmark28) solutions to the cloud computing paradigm. The basic objective of TNC from the perspective of endpoint integrity is to deny those network accesses to endpoints that do not meet certain minimum security criteria [[4–6]](#_bookmark29). In the network access control research, many related studies have been conducted in the ﬁeld of improving network connection protocol and improving TNC architecture.

* 1. *Trusted network connection protocol*

Luo et al. [[7]](#_bookmark31) followed the TNC framework to design an improved network access control system, T-NAC, which empha- sized the platform authentication and communication security. A network access control system based on the network

⇑ Corresponding author. Tel.: +86 13439195095; fax: +86 01067391742.

*E-mail address:* [laiyingxu@bjut.edu.cn](mailto:laiyingxu@bjut.edu.cn) (Y. Lai).

<http://dx.doi.org/10.1016/j.simpat.2014.10.011> 1569-190X/© 2014 Elsevier B.V. All rights reserved.

processor platform IXP2400 was designed. Latze et al. [[8]](#_bookmark32) proposed a strong bidirectional authentication protocol, which was based on TPM (Trusted Platform Module) and EAP-TLS authentication method. Latze et al. [[9]](#_bookmark33) later proposed a new authen- tication protocol: EAP-TPM. The ﬁrst EAP-TPM system was built in Switzerland. Then Latze et al. [[10]](#_bookmark34) improved EAP-TPM protocol based on zero-authentication. However, these protocols did not address the problem of Man-in-the-middle attacks. Wang et al. [[11]](#_bookmark35) analyzed the D–H (Difﬁe–Hellman) keys exchange protocol, and proposed a digital signature and signature veriﬁcation end-to-end protocol to solve the problem of Man-in-the-middle attacks. Yu et al. [[12]](#_bookmark36) studied the platform anon- ymous identity management defects of TNC and proposed a new method to improve these impairments. The new trusted certiﬁcation generation method was based on ID (Identity) encryption mechanism and improved DAA (Direct Anonymous Attestation) protocol. The protocol was more ﬂexible and security to manage identity on terminal platforms.

* 1. *Trusted access model*

To make the trusted network access mechanism more practicable, researchers proposed different solutions that focused on TNC architecture. Jungbauer and Pohlmann [[13]](#_bookmark37) proposed a method to determine the integrity of endpoints which served as a basis for trustworthy communication. The model did not require speciﬁc hardware such as TPM (Trusted Platform Mod- ule) or special operating system structure. It also supported exiting network infrastructures. Rehbock and Hunt [[14]](#_bookmark38) pro- posed a protocol stack that enabled the use of TNC in web-based environments and changed the TNC architecture to ensure additional security. The potential use of the TPM functionality within the TNC framework and experiences were given by Bente [[15]](#_bookmark39) and Heldenin [[16]](#_bookmark40), respectively. They further deﬁned a conceptual model for client-side policies that was based upon TNC’s IF-M (Interface-Measurement) protocol and showed that many policies can be enforced by extending the stan- dard TNC framework [[17]](#_bookmark41). Tang et al. [[18]](#_bookmark42) proposed a trusted network model based on the TPM, through which a trusted chain from terminals to network was established. Zhang [[19]](#_bookmark43) designed a TNC security model based on UCF (Universally Com- posable Framework) that can be extended to describe more security properties, such as anonymity. The model can be applied to analyze more protocols in the TNC architecture.

Cloud security is a system engineering problem. It requires additional security features not only for the endpoint security techniques, but also for the switches and routers, which are the core network equipment of Ethernet. In general, the existing TNC framework can be used to assess and verify end clients’ system state, but cannot be applied to network equipment directly. There are certain differences between network equipment and the endpoints to join in a network. Network equip- ment undertakes forwarding packets within routing or switching function in the network. They are service providers, whereas, endpoints use network only, and they are service clients. If accessed network equipments failure, it will affect the part of the cloud network, whereas, a failure endpoint only affects itself. Thus, network equipment pays more attention on trusted boot up process and trusted network services than terminals do. Trusted boot up process ensures network equip- ments static trust, trusted network service ensures its dynamic trust.

On the other hand, existing authentication protocols of network equipment have various weaknesses and are difﬁcult to be applied in the trusted network. For example, PAP (Password Authentication Protocol) is often used in router access authentication. However, PAP is not a strong and effective method of authentication. The password is transported in plain text. CHAP (Challenge Handshake Authentication Protocol) [[20]](#_bookmark44) is another router authentication protocol. The protocol is more secure than PAP. In CHAP, (1) the remote access server sends a challenge to the remote client that consists of a session ID and an arbitrary challenge string. (2) The remote client must return the user name and a Message Digest 4 (MD4) hash of the challenge string, the session ID, and the MD4-hashed password. (3) The authenticator checks the response against its own calculation of the expected hash value. If the values match, the authentication is acknowledged; otherwise the connec- tion SHOULD be terminated. (4) At random intervals, the authenticator sends a new challenge to the peer, and repeats steps 1–3. However, CHAP [[21]](#_bookmark45) also has its weaknesses: (1) In the authentication server, the user’s password was stored in plain text, which provided opportunities for intruders to obtain a user’s password. (2) The protocol supports one-way authentica- tion only. (3) The user password in CHAP is shared between two communication parties, and thus keys distribution and updating can cause inconsistency problems. (4) If a user used a simple password, the protocol could not prevent the dictio- nary attack. (5) In order to prevent the insertion channel attacks, the authentication server must reprint certiﬁcation peri- odically. If the cycle time interval is too long, it can give the intruder opportunities.

This paper presents a network device access authentication protocol. In our protocol, in addition to equipment’s platform authentication, the administrator also should be authenticated. The identiﬁcation authentication process is actually the binding administrator to the trusted platform, to ensure the administrator is legal on trusted platform. Our protocol is to avoid malicious behavior from illegal administrator login trusted platform or legitimate user login illegal platform. The authentication protocol achieves these targets: safety, credibility, and low overhead. The security of the protocol is analyzed by a formalization method based on BAN (Burrows–Abadi–Needham) logic, which can reveals vulnerabilities and redun- dancy [[22]](#_bookmark46). The protocol processes are formalized with HLPSL (High-Level Protocol Speciﬁcation Language) [[23]](#_bookmark47). The formal- ization of the protocol processes is tested by plugging into an attacking model of the safety testing tool to check whether the protocol is secure or not.

In the rest of the paper, Section [2](#_bookmark4) shows the design of the authentication method for the trusted network devices to join in the network. Section [3](#_bookmark13) formalizes our protocol by BAN logic for safety analysis. Section [4](#_bookmark14) presents the experiment, which uses the attack model to attack the protocol and to demonstrate that protocol is secure. Section [5](#_bookmark22) gives the network equip- ment performance evaluation. We conclude the paper in Section [6](#_bookmark30).

1. Design of the authentication protocol

As shown in [Fig. 1](#_bookmark6), the proposed Trusted Network Equipment Access Authentication Protocol (TNEAAP) consists of three major components: requester, boundary network equipment, and network authentication management server.

1. Requester (R): it is a piece of network equipment, in which a Trusted Platform Module (TPM) is embedded. The Requester sends the metric information to the authentication management server when it wants to be a member of the network.
2. Boundary network equipment (BNE): it is a strategy execution device that controls the requester to access the network.
3. Network authentication management server (NAMS): it is a decision builder that manages network equipment and an administrator in the trusted network.
   1. *Authentication process*

The proposed authentication process of TNEAAP is divided into the following three steps:

* + 1. Issuing of trusted certiﬁcation: A requester R registers itself to the trusted certiﬁcate server, as shown in [Fig. 1](#_bookmark6), and then R applies for a trusted certiﬁcate from the server. R sends hardware information, software information, operating system version, and public key of R to the trusted certiﬁcate server. The server veriﬁes information. If it is correct, the server will generate a trusted certiﬁcate and issue it to R.
    2. Platform authentication: This step performs the equipment platform authentication. R sends the Storage Measure- ment Log (SML) to the NAMS. NAMS validates whether SML accords with the trusted metric accessing rules. If infor- mation is certiﬁed by NAMS, the requester will be allowed to join into the network.
    3. Administrator authentication: When an administrator logs onto the network device (requester R), R sends the admin- istrator information to NAMS, which checks whether the administrator is legal or not.

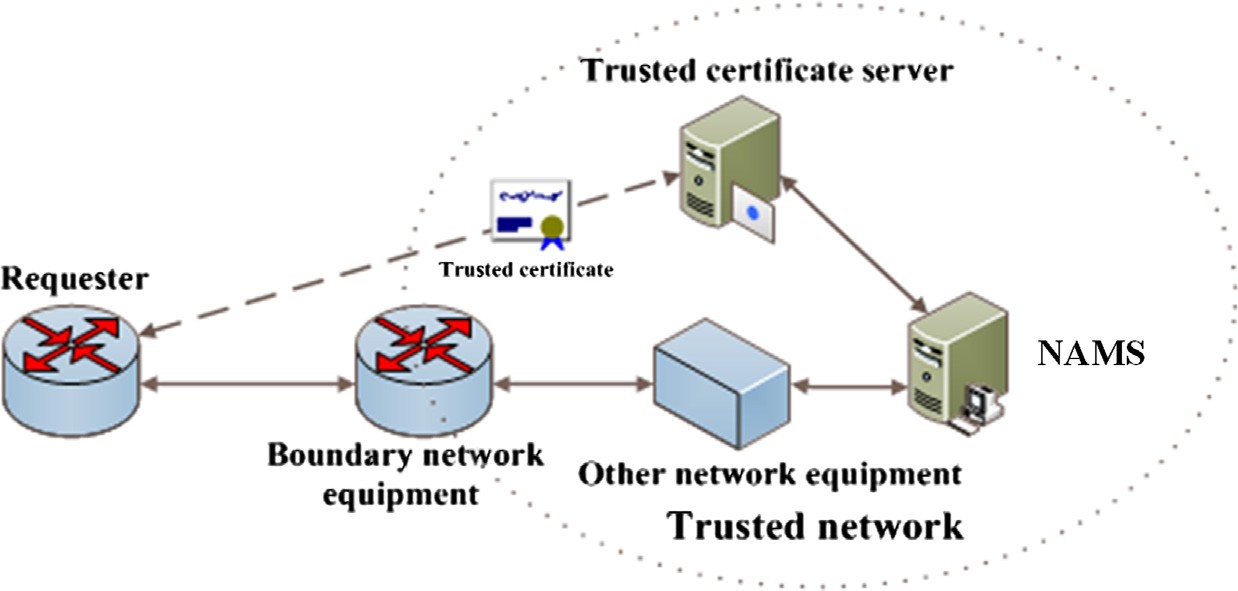
In this paper, we do not discuss the trusted certiﬁcate server’s strategy, and we focus on the security issues of the pro- posed authentication protocol.

* 1. *Formalization of TNEAAP*

The foregoing platform authentication and administrator authentication steps of TNEAAP are shown in [Fig. 2](#_bookmark7), which elab- orates the authentication process. The steps of the process listed in the ﬁgure are explained as follows.

* + 1. Platform authentication
       1. R sends a request to NAMS. R signs the NAMS’s public key NAMSpub and R’s platform identity plat\_ID by applying the private key of R (R—1), and R sends the signed message to the server NAMS.

*R* ! *NAMS :* f*plat ID*k*NAMSpub*g*R*—1 ð1Þ



* + - 1. NAMS sends a random number to R. In the step (i), NAMS has obtained R’s ID and NAMSpub, which means that R trusts NAMS. Then NAMS generates a random number nonceNAMS. NAMS sends nonceNAMS and Rpub signed by the private key of NAMS (NAMS—1) to R, where Rpub is the public key of R.

Fig. 1. TNEAAP architecture.

**R**

**NAMS**

1. **Authentication Request:**

**{plat\_ID||NAMS}R-1**

1. **Authentication Random Number:**

**{nonceNAMS||Rpub}NAMS-1**

**Platform**

**Authenticatio {{{PCR||nonceNAMS}R-1||SML||plat\_ID}NAMSpub}R-1**

**iii) Platform Authentication Information:**

**n**

**iv)The Authentication Pass:**

**{{Rsucc}R-1}NAMS-1**

**Platform Authenticatio n**

**User Identity**

1. **User Identity Authentication Request:**

**{SID||plat\_ID||user\_ID}R-1**

1. **Session Key Random Number:**

**{{nonceNAMS-SK}Rpub||SID}NAMS-1**

**User Identity**

**Authenticatio {{nonceR-SK}NAMSpub||SID||CMAC||{user\_ID||user\_PWD}SK}R-1 Authenticatio**

**vii) User Identity Information:**

**n**

**n**

**viii) The Authentication Pass:**

**{{R-NAMSsucc}SK}NAMS-1**

Fig. 2. Platform authentication and administrator authentication of TNEAAP.

*NAMS* ! *R :* f*nonceNAMS*k*Rpub*g*NAMS*—1 ð2Þ

* + - 1. R sends a platform authentication message to NAMS. R obtains the random number nonceNAMS and Rpub, which indi- cates that NAMS knows the public key of R. The hash digest of boot up processes is stored in PCR (Platform Conﬁg- uration Register), which cannot be deleted or modiﬁed. PCR’s value and nonceNAMS are signed by R’s private key R—1, that is, N = (PCR, nonceNAMS)R—1. R sends N, SML, and plat\_ID signed by R—1 to NAMS.

*R* ! *NAMS :* fff*PCR*k*nonceNAMS*g*R*—1k*SML*k*plat ID*g*NAMSpub*g*R*—1 ð3Þ

* + - 1. NAMS sends an acknowledgment message to R. NAMS unpacks the received message by Rpub and NAMS—1, obtains the PCR’s value. NAMS compares the hash digest of SML with PCR’s value. If the result is consistent, SML is not tam- pered. Then NAMS considers the boot up process is trusted. NAMS sends ACK (Acknowledgment) of platform authentication success to R and allows R to access network.

*NAMS* ! *R :* ff*ACK*g*Rpub*g*NAMS*—1 ð4Þ

* + 1. Administrator authentication (requires negotiation session key)

1. R sends a user authentication request to NAMS. The request includes plat\_ID, username (user\_ID), and the current session identiﬁer (SID). R sends the request signed by R—1 to NAMS.

*R* ! *NAMS :* f*SID*k*plat ID*k*user ID*g*R*—1 ð5Þ

1. NAMS generates and sends a session key random number to R. NAMS determines whether the user is legal and valid through the plat\_ID and user\_ID. If the information is valid, NAMS generates a session key random number nonce- NAMS-SK and encrypts it by Rpub. NAMS signs the cryptograph and SID, then sends this signature information to R.

*NAMS* ! *R :* f*nonceNAMS*-*SK* g*Rpub*k*SID*g*NAMS*—1 ð6Þ

1. R sends user identity information to NAMS. R decrypts SID by the public key of NAMS and the random number non- ceNAMS-SK using its own private key. Then, R generates a new random number nonceR-SK and message authentication codes CMAC, *CMAC* = *HMAC* — *SHA*1*SK*(*nonceNAMS-SK*, *SID*), which ensures information integrity. R uses the pseudo-ran- dom number generation function PRGF to calculate this session key SK using the nonceR-SK, nonceNAMS-SK and the current session identiﬁer SID. SK’s calculation equation is *SK* = *PRGF*(*nonceNAMS-SK*, *nonceR-SK*, *SID*). R encrypts the username user\_ID, user password user\_PWD, and trusted platform authentication certiﬁcate *R*cer by SK. R’s private key signs this cryptograph, {*nonceR-SK*}*NAMSpub*, SID and CMAC, sends this signature information to NAMS.

*R* ! *NAMS :* ff*nonceR*-*SK* g*NAMSpub*k*SID*k*CMAC* kf*user ID*k*user PWD*k*Rcer*g*SK*g*R*—1 ð7Þ

1. NAMS sends an acknowledgment message to R. NAMS—1 decrypts {*nonce**R-SK*}*NAMSpub*. NAMS receives the random number nonceR-SK. NAMS calculates the session key SK using the two existing random numbers of session and the current session identiﬁer SID. Then {*user*\_*ID*||*user*\_*PWD*||*Rcer*}*SK* is decrypted by the session key SK. NAMS can obtains the user name and password, compares the user information with the user registration information in NAMS. If the user is legal, NAMS sends ACK0 acknowledgment of administrator authentication success to R and allows the R to access the trusted network with the administrator login. The certiﬁcation process is completed with the following action:

*NAMS* ! *R :* ff*ACK*g*SK*g*NAMS*—1 ð8Þ

1. TNEAAP formalization analysis

According to BAN logic analysis method and the characteristics of TNEAAP, we analyze whether the authentication pro- tocol is secure or not.

* 1. *Analysis procedure*

The analysis procedure is as follows:

* + 1. According to BAN logic representation, the authentication protocol is formalized.
    2. The security goals of the protocol are determined. The assumptions are initialized. The security goals and initial assumptions are described by logical symbols.
    3. The proof of security is to show that if the initial assumptions are applied to the messages of the protocol, and all of the security goals can be inferred from the messages.
    4. In the reasoning process, the protocol defects and redundancy can be detected.

The authentication protocol is formalized as Eqs. [(1)–(8)](#_bookmark5), and the other analysis steps are as follows.

* 1. *Security targets*

This section discusses the general BAN logic forms of security objectives. The primary goal is *A*| *X* and *B*| *X*. The ulti- mate goal is *A*| *B*| *X* and *B*| *A*| *X*.

÷ ÷ ÷ ÷

÷ ÷

The initialization assumption is the primary goal in BAN. Thus, the security goals can be deﬁned as the following seven goals:

*R*j÷ *NAMS*j÷ j

*Rpub*

# ———!

*R* ð9Þ

*NAMS*j÷ *R*j÷ j *NAMSpub NAMS* ð10Þ

*NAMS*j÷ *R*j÷ *P*—*C*—*R*———!

# ð11Þ

*R*j÷ *NAMS*j÷ *ACK* ð12Þ

*R*j÷ *NAMS*j÷ *NNAMS*-*SK* ð13Þ

*NAMS*j÷ *R*j÷ *NR*-*SK* ð14Þ

*R*j÷ *NAMS*j÷ *ACK*0 ð15Þ

In words, the seven goals are:

r R believes that NAMS believes that Rpub is R’s public key.

s NAMS believes that R believes that NAMSpub is NAMS’s public key.

t NAMS believes that R believes that PCR is credible.

u R believes that NAMS believes that ACK is credible.

v R believes that NAMS believes that nonceNAMS-SK (NNAMS-SK) is credible.

w NAMS believes that R believes that nonceR-SK (NR-SK) is credible.

x R believes that NAMS believes that ACK0 is credible.

The fourth goal is to prove that the platform authentication process is trusted. The seventh goal is to prove that the administrator authentication process is trusted. These two goals are the most important ones.

* 1. *Initialization assumption*

According to the characteristics of TNEAAP, this paper makes the following initialization assumptions:

*R*j÷j *NAMSpub NAMS* ð16Þ

*R*j÷ *N*—*A*—*M*——*S*—! *N*

j)

*NAMS*

ð*nonce*

*NAMS*

Þ; *N*

*NAMS*-*SK*

# ð17Þ

*NAMS*j÷ *R*j) *NR*-*SK* ð18Þ

*R*j÷ *#*ð*NNAMS*; *NNAMS*-*SK* ; *Rpub*Þ ð19Þ

*NAMS*j÷ *#*ð*PCR*; *NR*-*SK* Þ ð20Þ

*NAMS*j÷j *Rpub R* ð21Þ

———!

* 1. *Analysis*

Logical Reasoning Process 1: From the Eq. [(2)](#_bookmark8) we can derive:

*R*j÷j *NAMSpub NAMS*; *R* . f*N* ; *R*

—————

g*NAMS*—1 ð22Þ

! *NAMS NAMS*

We can infer the following:

*R*j÷ *NAMS*j~ *NNAMS*; *R*j÷ *NAMS*j~ *Rpub*; *R*j÷ *#*ð*Rpub*Þ ð23Þ

Then, we draw the conclusion:

*R*j÷ *NAMS*j÷ j

———!

*Rpub R* ð24Þ

Logical Reasoning Process 2:

Similarly to reasoning process 1, from the Eq. [(2)](#_bookmark8) we can draw the conclusion.

*NAMS*j÷ *R*j÷ j *NAMSpub NAMS* ð25Þ

—————!

Logical Reasoning Process 3: From the Eq. [(3)](#_bookmark9), we can obtain

———!

*NAMS* . fff*PCR*; *N*

We can then infer:

*NAMS*

g*R*—1; *SML*; *plat ID*g*Ks*g*R*—1; *NAMS*j÷ j

*Rpub R* ð26Þ

*NAMS* . ff*PCR*; *N*

We can also infer:

*NAMS*

g*R*—1; *SML*; *plat ID*g*NAMSpub*

*NAMS*

; *NAMS*j÷j —————! *NAMS* ð27Þ

*pub*

———!

and

*NAMS* . ff*PCR*; *N*

*NAMS*

g*R*—1; *SML*; *plat ID*g; *NAMS*j÷j *Rpub R* ð28Þ

*NAMS*j÷ *R*jf*PCR*; *NNAMS*g; *NAMS*j÷ *#*ð*PCR*Þ ð29Þ

We draw the conclusion

*NAMS*j÷ *R*j÷ *PCR* ð30Þ

Logical Reasoning Process 4: From the Eq. [(4)](#_bookmark10), we can infer:

*R*j÷ *NAMS*jf*R*

*succ*

g*Rpub*

; *R*j÷ *#*ð*R*

*succ*

*R*

# Þ; *R*j÷j ———! *R* ð31Þ

*pub*

And draw the conclusion

*R*j÷ *NAMS*j÷ *ACK* ð32Þ

Logical Reasoning Process 5:

From the Eq. [(6)](#_bookmark11), we can infer:

*R*j÷ *NAMS*j~ f*SID*; f*NNAMS*-*SK* g*Rpub*g; and ð33Þ

*R*j÷ *#*ð*N*

*NAMS*-*SK*

*R*

# Þ; *R*j÷j ———! *R* ð34Þ

*pub*

We draw the conclusion

*R*j÷ *NAMS*j÷ *NNAMS*-*SK* ð35Þ

Logical Reasoning Process 6:

Similarly to reasoning 5, we can draw the conclusion from the Eq. [(7)](#_bookmark12):

*NAMS*j÷ *R*j÷ *NR*-*SK* ð36Þ

Logical Reasoning Process 7:

From the Eq. [(8)](#_bookmark13), we can infer:

*R*j÷ *NAMS*j~ f*ACK*0g*R*; and ð37Þ

*R*j÷ *#*ð*ACK*0Þ; *R*j÷j *Rpub R* ð38Þ

———!

And we draw a conclusion

*R*j÷ *NAMS*j÷ *ACK*0 ð39Þ

* 1. *Results*
     1. Although PCR, SML, platform ID, ACK, and ACK0 are not responsible for the authentication procedure directly, they are not redundant information. They are important for the security of the trusted network, and they play determinative roles in TNEAAP. We do not ﬁnd redundant information in TNEAAP based on aforementioned analysis, thus it is a con- cise protocol.
     2. In this method, seven security targets can be inferred from the messages of R and NAMS. Thus, TNEAAP is secure in BAN logic reasoning.

1. The security testing and analysis of TNEAAP

In this section, we use an experiment to further demonstrate the protocol is secure through an attacking detection method. The experiment should ﬁnd defects of TNEAAP if there exists any. The protocol authentication processes are formal- ized using HLPSL [[24]](#_bookmark48) protocol description language. Then formalization of the protocol process is plugged into an attacking model of a safety testing tool. Finally the tool gives a conclusion whether TNEAAP can prevent the attack generated by the tool.

* 1. *Security targets*

In this section, we choose Dolev-Yao intruder model [[25]](#_bookmark49) for testing our protocol. In this model, an intruder controls the whole network and can perform any operations. The intruder can intercept, analyze, and revise all of messages. The intruder can pretend to be any agent and send disguised messages to anyone in the network. The attacker has the following knowl- edge and abilities:

* + 1. The attacker is familiar with encryption, decryption, hash, and other cryptographic operations. The attacker has its own public key and private key.
    2. The attacker has obtained the subject’s identiﬁer and public key.

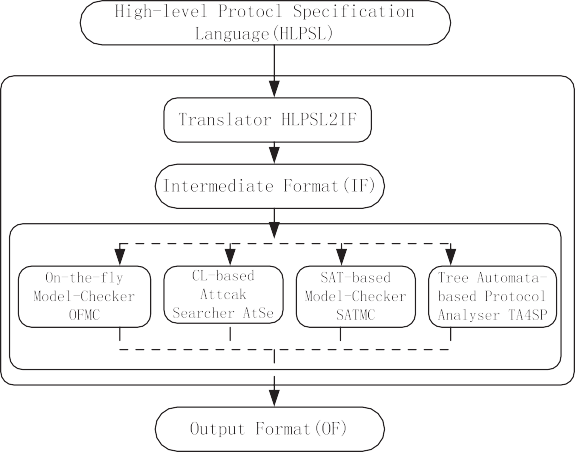


Fig. 3. Architecture of AVISPA.

Table 1

Deﬁnition of the basic roles.

Basic role Basic role conﬁguration

a role a (R, NAMS:agent, Ka, Kr:public\_key, SND, RCV:channel(dy))

r role r (R, NAMS:agent, Ka, Kr:public\_key, SND, RCV:channel(dy))

* + 1. The attacker has the knowledge and ability of cryptanalysis.
    2. The attacker can make various attacks, such as knowledge and ability of replay attacks.

This is the strongest intruder model. If TNEAAP can meet the security goals under this intruder model, it proves the pro- tocol can prevent all possible attacks from any attacker.

In this experiment, we deﬁne 9 security goals:

1. The transportation process of PCR is conﬁdential.
2. The transportation process of SML is conﬁdential.
3. The transportation process of random number nonceNAMS is conﬁdential.
4. The transportation process of session key random number nonceNAMS-SK is conﬁdential.
5. The transportation process of session key random number nonceR-SK is conﬁdential.
6. The transportation process of the administrator’s password is conﬁdential;
7. The transportation process of the administrator’s username is conﬁdential.
8. The platform authentication success ACK is correct.
9. The user authentication success ACK0 is correct.

According to the above security goals, we use AVISPA (Automated Validation of Internet Security Protocols and Applica- tions) [[26]](#_bookmark50) network communication protocol security inspection system to test the security of our protocol.

* 1. *AVISPA*

AVISPA is a set of the establishment and analysis tools of security protocols [[26]](#_bookmark50). It is one of the widest used protocols in the cryptography. As shown in [Fig. 3](#_bookmark15), it combines four types of analyses at backends: On-the-ﬂy Model-Checker (OFMC), CL- based Attack Searcher (CL-AtSe), SAT-based Model-Checker (SATMC), and Tree-Automata based Protocol Analyzer (TA4SP). The user ﬁrst inputs the participants’ identiﬁcation of protocol and then selects the running environment, goal, attacker abil- ity variables. Finally, the user speciﬁes the desired security properties in order to ﬁnd whether there are problems in the protocol under test. The code written in HLPSL language is translated into IF (Intermediate Format) language through HLPSL2IF translation tools. Analysis of terminal AVISPA tool set can directly read IF language. It can analyze whether security goals are successful or not. If the protocol is unsafe, the analysis terminal will give us the attack track events.

* 1. *Experiment process*
     1. *Basic roles*

In HLPSL, each participant is deﬁned in a module separately, called basic role, which describes its initial state and its state transitions. We deﬁne two basic roles, as shown in [Table 1](#_bookmark16): a network authentication manager server NAMS (a) and a Requester (r).

* + 1. *Security goals*

For evaluating the security properties of TNEAAP, we ﬁrst formulize its security goals [[27]](#_bookmark51). The speciﬁed equations are used to assess whether the goals are achieved or not.

AVISPA deﬁnes different macros for formalizing the security goals. In our experiment, the macros are described as follow:

Table 2

Summary of session conﬁgurations.

Scenario Session conﬁguration

Scenario 1 Session(a, r, ka, kr)

Scenario 2 Session(a, i, ka, ki)

Scenario 3 Session(i, r, ki, kr)

1. Macro of information secrecy. T is the information produced by agent A. If T is a shared secret and is shared between agent A and a group of agents, B and C, the secrecy of information T is expressed as follows:

# SecretðT; t; A; B; CÞ

where t is the identiﬁcation of the information T.

1. Macro of strong authentication detection. This property is formalized using two macros as follows:

# RequestðB; A; t; TÞ

WitnessðA; B; t; TÞ

where Request (B, A, t, T) indicates that agent B receives information T (identiﬁed by t) from agent A. Witness (A, B, t, T) indi- cates that agent A receives information T (identiﬁed by t) from agent B.

For evaluating the security of TNEAAP, the following security goals are deﬁned:

1. Authentication of ACK and ACK0 , where their process IDs are r\_ack1 and r\_ack2, respectively. We model this goal in HLPSL. Role r (R, NAMS: agent, Ka, Kr:public\_key, SND, RCV:channel(dy))

Played\_by R init State:=0 Transition:

State0 :=6/ request(R, NAMS, r\_ack10 ) State0 :=10/ request(R, NAMS, r\_ack20 )

n

n

It means that R requests to check both ACK and ACK0 . If the authentication is successful, R can obtain an acknowledge message.

1. Check of the SK shared between NAMS and R. SK is only known by NAMS and R. Similar to SK, check of the PCR, SML, nonceNAMS, user\_ID, user\_PWD, nonceNAMS-SK and nonceR-SK are all needed. These goals are modeled in HLPSL as follows:

Role a (R, NAMS: agent, Ka, Kr:public\_key, SND, RCV:channel(dy))

Played\_by NAMS init State:=0 Transition:

State0 :=5/ request(PCR0 , a\_pcr, {R, NAMS})

n

/ request(SML0 , a\_sml, {R, NAMS})

n

/ request(Nonce10 , a\_nonce1, {R, NAMS}) State0 :=9/ request(User\_id0 , a\_userid, {R, NAMS})

n

n

/ request(User\_pwd0 , a\_userpwd, {R, NAMS})

n

/ request(Nonce30 , a\_nonce3, {R, NAMS})

n

/ request(SK0 ,a\_sk, {R, NAMS})

n

Role r (R, NAMS: agent, Ka, Kr:public\_key, SND, RCV:channel(dy))

Played\_by R init State:=0

State0 :=8/nrequest(Nonce20 , a\_nonce2, {R, NAMS})

Table 3

Summary of test outputs from OFMC backend, and from CL-AtSe backend.

SUMMARY SUMMARY

SAFE SAFE

DETAILS DETAILS

BOUNDED\_NUMBER\_OF\_SESSIONS BOUNDED\_NUMBER\_OF\_SESSIONS

PROTOCOL TYPED\_MODEL

D:nNPLABntempn130401203137024905.if PROTOCOL

GOAL D:nNPLABntempn130401203858035709.if

As\_speciﬁed GOAL

BACKEND As Speciﬁed

OFMC BACKEND

COMMENTS CL-AtSe

STATISTICS STATISTICS

parseTime: 0.00 s Analyzed: 4 states

searchTime: 0.03 s Reachable: 4 states

visitedNodes: 4 nodes Translation: 0.01 s

depth: 2 plies Computation: 0.00 s

Table 4

Summary of CHAP from OFMC backend, and from CL-AtSe backend.

SUMMARY SUMMARY

UNSAFE UNSAFE

DETAILS

ATTACK\_FOUND DETAILS

PROTOCOL ATTACK\_FOUND

D:nNPLABntempn130570765157676751.if TYPED\_MODEL

GOAL PROTOCOL

secrecy\_of\_sec\_kab1 D:nNPLABntempn130570766808871194.if BACKEND

OFMC GOAL

COMMENTS Secrecy attack on (kab)

STATISTICS

parseTime: 0.00 s BACKEND

searchTime: 0.01 s CL-AtSe

visitedNodes: 1 nodes

depth: 1 plies STATISTICS

ATTACK TRACE

i ->(a,3): start Analyzed: 7 states

(a,3) -> i: a Reachable: 7 states

i -> (a,3): x238 Translation: 0.01 s

(a,3) -> i: Na(2).h(kab.Na(2).x238.a) Computation: 0.00 s i -> (i,17): kab

i -> (i,17): kab

ATTACK TRACE

% Reached State: i -> (a,3): start

%% secret(kab,sec\_kab1,set\_61) (a,3) -> i: a

% contains(a,set\_61)

% contains(b,set\_61) i -> (a,3): Nb(2)

% state\_chap\_Init(b,i,kbi,h,0,dummy\_nonce,dummy\_nonce,set\_77,9) (a,3) -> i: n2(Na).{kab.n2(Na).Nb(2).a}\_h

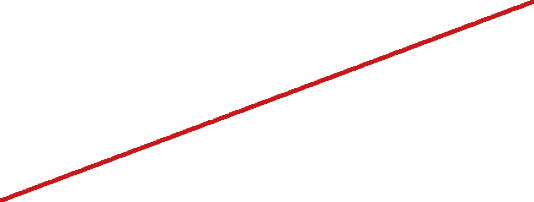
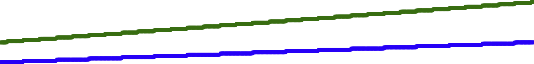
% state\_chap\_Init(a,i,kai,h,0,dummy\_nonce,dummy\_nonce,set\_74,6) & Secret(kab,set\_61); Witness(a,b,na,n2(Na)); Add a to set\_61;

% state\_chap\_Init(a,b,kab,h,2,Na(2),x238,set\_61,3) & Add b to set\_61;

% state\_chap\_Resp(b,a,kab,h,0,dummy\_nonce,dummy\_nonce,set\_69,3)

% witness(a,b,na,Na(2))

3.5



n=50

n=10

n=5

3

2.5

Storage overhead

2

1.5

1

0.5

00 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

Paket Loss Probability

Fig. 4. Storage overhead.

In the model NAMS declares PCR, SML, nonceNAMS, user\_ID, user\_PWD, nonceR-SK and SK as the secrets, and R declares nonceNAMS-SK as a secret, where a\_x stands for x’s process id.

* + 1. *Session scenarios*

For the validation purpose, we deﬁne three different scenarios. First, we implement a single session with all the roles played by legitimate agents (Scenario 1). Then we test the situations in which the intruder would impersonate the network authentication manager server (Scenario 2) or the requester (Scenario 3). [Table 2](#_bookmark17) lists the HLPSL deﬁnition of the sessions associated with each of the mentioned scenarios, where kx refers to the public key of x.

Table 5

Computational overhead of cryptographic operations in Eqs. [(3) and (7)](#_bookmark9).

|  |  |  |
| --- | --- | --- |
| Type of operation | Eq. [(3)](#_bookmark9)  (calculation times) | Eq. [(7)](#_bookmark12)  (calculation times) |
| Symmetric key encryption | 0 | 1 |
| Symmetric key decryption | 0 | 0 |
| Asymmetric key encryption | 1 | 1 |
| Asymmetric key decryption | 2 | 1 |
| MAC | 0 | 1 |

* + 1. *Experiment results*

The experiments are conducted based on the aforementioned model speciﬁcation. For our veriﬁcation, we have used OFMC and CL-AtSe backends to search for the attacks on the protocol. The test outputs of the experiment results are sum- marized in [Table 3](#_bookmark18). The left column lists the output from the OFMC backend and the right column is from the CL-AtSe backend.

According to the summary results in [Table 3](#_bookmark18), TNEAAP is safe in both OFMC and AtSe backends (Summary: SAFE), and no vulnerabilities in the proposed protocol. If some defects are detected, Summary will be UNSAFE. DETAILS section provides the information that an attack is found in the protocol speciﬁcation. The IF form of the protocol resides in the path given under the PROTOCOL section of the output, with the ﬁle name, 130401203137024905.if. The GOAL section of output describes the result of the goal, which is written in the speciﬁcation for the veriﬁcation process. The backend that veriﬁes the protocol speciﬁcation is OFMC or CL-AtSe, which is given under the BACKEND. The STATISTICS gives us the time required to execute the protocol speciﬁcation by the tool and the number of the visited nodes or states during the execution.

For comparison, we have used OFMC and CL-AtSe backends to search for the attacks on the CHAP protocol. The test out- puts of the experiment results are summarized in [Table 4](#_bookmark19). The left column lists the output from the OFMC backend and the right column is from the CL-AtSe backend.

According to the summary results in [Table 4](#_bookmark19), CHAP is unsafe in both OFMC and AtSe backends (Summary: UNSAFE), DETAILS section provides the information that a Man-in-the-middle attack is found in the protocol speciﬁcation.

1. Network equipment performance evaluation

Like all security protocols, TNEAAP take extra steps to ensure the security of the network and its devices. This section evaluates the overhead of implementing TNEAAP from network equipment’s point of view. We take both storage and com- putation overheads into consideration.

* 1. *Storage overhead*

In TNEAAP, each requester has to store two keys permanently: the secret key SK and the NAMS’s public key NAMSpub. Additionally, each requester also needs to store two nonce values: nonceNAMS and nonceNAMS-SK. Considering the request-

ers that attempt to access a given node in a completely random basis at mean rate 1/T, the second step of authentication messages in Eq. [(2)](#_bookmark8) received by NAMS can be modeled as a Poisson process with mean 1/T. For each authentication message received, NAMS stores a nonceNAMS, until it receives the fourth step of authentication message in Eq. [(4)](#_bookmark10) from NAMS or until a timer set to TLifetime expires. Suppose that the TLifetime has an upper bound and is reached whenever a packet is lost, assuming a packet loss probability of the network is P, the average number of nonceNAMS that the requester must store is given by Eq. [(40)](#_bookmark24), where TACK denotes the average time elapsed between the reception of an Eq. [(2)](#_bookmark8) message and its corresponding Eq. [(4)](#_bookmark10) message. We assume that the storage overhead of nonceNAMS and nonceNAMS-SK is the same. Therefore, the requester storage overhead is

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table 6  Comparison with other protocols. |  | | | |
|  | Kerberos [[28]](#_bookmark52) | LEAP [[29]](#_bookmark53) | SPINS-based protocol [[30]](#_bookmark54) | TNEAAP |
| Authentication capability | Yes | No | Yes | Yes |
| Authorization capability | No | Two levels: legitimate member of the sensor | No | Yes |
| Storage cost | 1 symmetric key | network or attacker  1 symmetric key + identities of neighboring | 1 symmetric key | 1 public key + 1 symmetric |
| Computation overhead | 1 encryption + 2 | nodes  1 MAC + 1 pseudo-random function | 1 decryption + 2 | key + 2 nonce  2 encryption + 1 |
| Support trusted module | decryption No | No | MAC  No | decryption + 1 MAC Yes |

Storage overhead ¼ *N* × Σ1 *TLifetimeP* þ 1 *TACK* ð1 — *P*ÞΣ ð40Þ

*T*

*T*

As an example, we set TACK = 10 s, TLifetime = 2TACK = 20 s, and T = 5 min. [Fig. 4](#_bookmark20) shows overheads incurred from 5, 10, and 50 requesters, with an increasing packet loss probability. As can be seen in the ﬁgure, the average number of nonceNAMS and nonceNAMS-SK that NAMS stores is small. It implies that NAMS does not need to have a large buffer to maintain the nonce values.

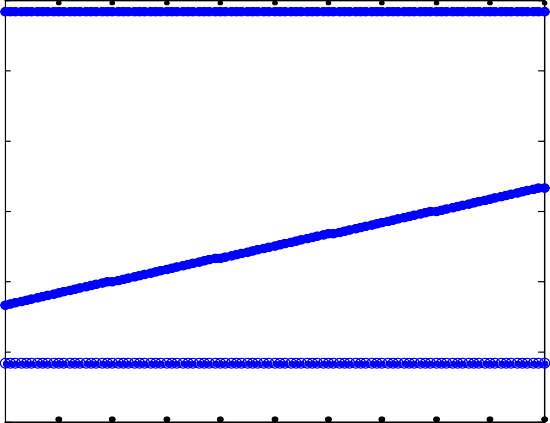
* 1. *Computation overhead*

The overall computational overhead incurred by TNEAAP is a ﬁxed number. The individual overheads are from Eqs. [(3)](#_bookmark9) [and (7)](#_bookmark9). [Table 5](#_bookmark21) summarizes the individual overheads in NEAAP for each type of requesters.

* 1. *Comparison with other authentication protocols*

Now we compare the capacity and overhead of TNEAAP with that of the other security protocols. The results are summa- rized in [Table 6](#_bookmark23). As shown in [Figs. 5](#_bookmark25) and [6](#_bookmark26), for 50 requesters, our method does not need to have a large buffer to maintain nonce values, and computational overhead is similar to the other protocols. However, TNEAAP supports more security func-

6



 LEAP

 TNEAAP

 SPINS

 Kerberos

5

4

Storage overhead

3

2

1

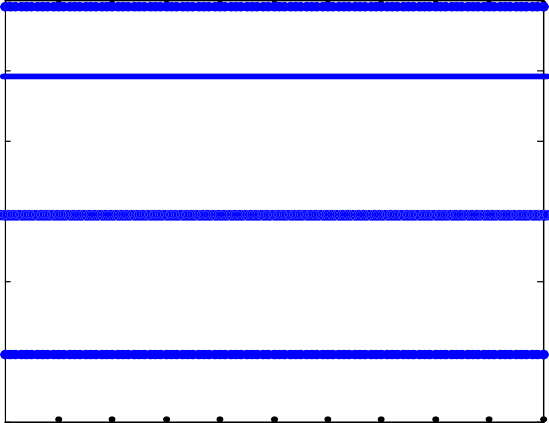
0

0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

Paket Loss Probability

Fig. 5. Comparison curve of storage overhead at *n* = 50.

45



 TNEAAP

 SPINS

 Kerberos

 LEAP

40

Computation overhead (times)

35

30

25

20

15

0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

Paket Loss Probability

Fig. 6. Comparison curve of computation overhead at *n* = 50.

tions, including both authentication capability and authorization capability, which ensure the credibility of the network sys- tem is strong.

1. Conclusions

Current trusted network access authentication research pays major attentions to the terminals, and thus the TNC (Trusted Network Connection) does not have an authentication protocol for network devices accessing. In this paper, we developed the TNEAAP protocol, which is more suitable for network equipment, more secure, and more reliable, due to the develop- ment of additional mechanism for the equipment’s platform authentication.

In the study of TNEAAP, we theoretically and experimentally veriﬁed three issues: (1) TNEAAP does not contain any unnecessary information. We analyzed the redundant information by BAN logical system. The results show that the protocol is a concise protocol. (2) The protocol is security in theory. The BAN logic safety analysis has proved TNEAAP is safe and reli- able. (3) The protocol is secure in experiment. We tested the protocol under the strongest attack model in our experiments. The nine goals which decided the protocol’s security have been achieved in the attack model.

Acknowledgments

This research is partly supported by Funding Project for Academic Human Resources Development in Institutions of Higher Learning under the Jurisdiction of Beijing Municipality (PHR201108016), Open Research Fund of Beijing Key Labora- tory of Trusted Computing.

References

1. [Y. Chen, W.T. Tsai, Service-Oriented Computing and Web Software Integration, fourth ed., Kendall Hunt Publishing, 2014](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0005).
2. [A. Ioannis Moschakis, H.D. Karatza, Evaluation of gang scheduling performance and cost in a cloud computing system, J. Supercomput. 59 (2) (2012)](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0010) [975–992](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0010).
3. TCG Trusted Network Connect Work Group. TNC Architecture for Interoperability (April 2008), <[http://www.trutedcomputinggroup.org/resources/](http://www.trutedcomputinggroup.org/resources/tnc_architecture_for_interoperability_version_13) [tnc\_architecture\_for\_interoperability\_version\_13](http://www.trutedcomputinggroup.org/resources/tnc_architecture_for_interoperability_version_13)> (Speciﬁcation Version 1.3 Revision 6).
4. Trusted Computing Group, TPM main speciﬁcation, Version 1.2 rev. 85 <[http://www.trustedcomputinggroup.com](http://www.trustedcomputinggroup.com/)>, 2011.4.
5. Trusted Computing Group, TCG Storage Architecture Core Speciﬁcation, Version 1.2 <[http://www.trustedcomputinggroup.com](http://www.trustedcomputinggroup.com/)>, 2007.
6. Trusted Computing Group, TCG Trusted Network Connect TNC Architecture for Interoperability Speciﬁcation, Version 1.2 <[http:// www.trustedcomputinggroup.com](http://www.trustedcomputinggroup.com/)>, 2007.
7. A. Luo, C. Lin, Z. Chen, X. Peng, et al., TNC-compatible NAC system implemented on network processor, in: The 32nd IEEE Conference on Local Computer Networks, 2007, Dublin, Ireland, pp. 1069–1075.
8. C. Latze, U. Ultes-Nitsche, F. Baumgartner, Strong mutual authentication in a user-friendly way in EAP-TLS: software, telecommunications and computer networks, in: 15th International Conference on SoftCOM, 2007, Dubrovnik, Croatia, pp. 27–29.
9. C. Latze, U. Ultes-Nitsche, A proof-of-concept implementation of EAP-TLS with TPM support, in: Proceedings of International Conference on Citeseer, 2008, pp. 1–12.
10. C. Latze, U. Ultes-Nitsche, F. Baumgartner, Towards a zero conﬁguration authentication scheme for 802.11 based networks, in: Proceedings of International Conference on IEEE, 2008, pp. 367–373.
11. [Y. Wang, Z. Li, P. Yu, et al, Improved trusted network connect protocol to prevent middle attack, Appl. Res. Comput. 27 (2010) 4309–4311](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0055).
12. [A. Yu, X. Chu, D. Feng, et al, Research of platform anonymous identity management based on trusted chip, Chinese J. Comput. 33 (2010) 1703–1712](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0060).
13. M. Jungbauer, N. Pohlmann, Integrity check of remote computer systems trusted network connect. ISSE/SECURE 2007 – Securing Electronic Business Processes: Highlights of the Information Security Solutions Europe/SECURE 2007 Conference, 2007, pp. 228–237.
14. [S. Rehbock, R. Hunt, Trustworthy clients: extending TNC to web-based environments, Comput. Commun. 32 (2009) 1006–1013](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0070).
15. I. Bente, J. von Helden, Towards trusted network access control, in: Proceedings of the First International Conference Future of Trust in Computing, 2008, pp. 157–167.
16. J. vonHelden, I. Bente, Towards real interoperable, real trusted network access control, in: ISSE 2008 Securing Electronic Business Processes, 2008, pp. 152–162.
17. I. Bente, J. Vieweg, J. von Helden, Privacy Enhanced Trusted Network Connect INTRUST 2009, LNCS 6163, 2010, pp. 129–145.
18. [J. Tang, S. Song, L. Zhao, et al, Trusted network model based on trusted platform module, Comput. Eng. 37 (11) (2011) 117–119](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0090).
19. Trusted Network Universally Composable, Scientia Sinica (Informations) 2(2) (2010) 200–215.
20. W. Simpson, PPP Challenge Handshake Authentication Protocol (CHAP), RFC1994, 1996.
21. [Y. Liu, F. Zhu, Q. Shi, An improved scheme of challenge handshake authentication protocol, Comput. Eng. 31 (5) (2005) 168–169](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0105).
22. [Y. Wang, J. Wang, M. Wang, et al, Security analysis of routing protocol for MANET based on BAN logic, J. China Inst. Commun. 25 (4) (2010) 125–129](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0110).
23. F. Dadeau, P.C. He´ am, R. Kheddam, Mutation-based test generation from security protocols in HLPSL, veriﬁcation and validation (ICST), in: 2011 IEEE Fourth International Conference on Software Testing, 2011, pp. 240–248.
24. [L.C. Paulson, Inductive analysis of the internet protocol TLS, ACM Trans. Inform. Syst. Secur. (TISSEC) 2 (3) (1999) 332–351](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0120).
25. Q. Chen, Automatic veriﬁcation of web service protocols for epistemic speciﬁcations under Dolev-Yao Model, in: 2010 International Conference on Service Sciences (ICSS), 2010, pp. 49–54.
26. <http://www.avispa-project.org/>.
27. [N. Toledo, M. Higuero, J. Astorga, M. Aguado, Design and formal security evaluation of NeMHIP: a new secure and efﬁcient network mobility management protocol based on the host identity protocol, Comput. Secur. 32 (2013) 1–18](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0135).
28. J. Kohl, C. Neuman, The Kerberos network authentication service (v5). RFC 1510, September, 1993.
29. [S. Zhu, S. Setia, S. Jajodia, LEAP+: efﬁcient security mechanisms for large-scale distributed sensor networks, ACM Trans. Sens. Networ. (TOSN) 2 (4) (2006) 500–528](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0145).
30. [A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, D. Culler, SPINS: security protocols for sensor networks, Wirel. Networ. 8 (5) (2002) 521–534](http://refhub.elsevier.com/S1569-190X(14)00162-2/h0150).