Towards Cooperative Self-Protecting Mobile Devices using Trustful Relationships

Stephan Groß
Technische Universität Dresden
Department of Computer Science
Institute for System Architecture
D-01062 Dresden, Germany
Email: Stephan.Gross@tu-dresden.de

***Abstract*— Security remains a major topic in today's networks. Especially in the mobile area there are many security issues that have not yet been satisfactorily addressed, let alone been solved. Most of the security mechanisms and techniques used in wired networks tend to rely upon a fixed network topology. For example, firewalls and Intrusion Detection Systems are placed at central points of a network and configured with a model of the network's structure to control and analyse the data flows transfered for harmful content. If at all, adopting these measures for the mobile world is not easy due to its dynamically changing environments and the mobile device's resource constraints that do not allow demanding analyse tasks. In this paper, we present our work in progress of developing a system architecture for cooperative and self-protecting mobile devices. Our approach is based on the assumption that a mobile network can be protected by securing all participating devices or at least all honest participating devices. Thus, we no longer insist on a global view of the network but on several local views. To make these views as precise as possible and to avoid serious performance bottlenecks on a single device we propose a strategy for coupling trusted mobile devices together into a cooperating group.**

I. Introduction

Like all networks today, mobile networks are exposed to a multitude of threats. As mobile devices are used in different possibly not trustworthy environments they are even more at risk than common ones. What is more, recent surveys have shown that mobile devices exceedingly endanger existing networks as they provide an opportunity for an attacker to avoid security defence lines like a firewall or a VPN gateway [1]. Thus, we have to put more effort into the development of secure mobile devices.

Today, we generally distinguish between preventive and supervisory measures to secure a computing device. Preventive measures try to avoid security threats by suppressing potential harmful actions. This can be realized by rather "simple" techniques to control the access to system resources (e.g. firewalls and anti-virus software) or by more sophisticated techniques like Virtual Private Networks utilizing cryptographic means to protect network traffic from wiretapping. On the other hand supervisory measures are used to detect abnormal system behaviour. For example log monitors check the system files if any unwanted action has happened in the past. This includes the actions of preventive measures, e.g. if the traffic of a specific network address was correctly filtered by a firewall. Common Intrusion Detection Systems (IDS) are able to detect potential harmful operations even at the time when they arise. If they are proactive we speak of Intrusion Detection and Response Systems. Unfortunately, the setup and configuration of these mechanisms is not trivial. On the contrary, one needs a profound security knowledge for these tasks. What is more, there is a need for constant maintenance and surveillance during operation. Altogether this is far beyond the abilities of a mobile device's common user. Far too often he is even not aware of possible security threats. Consequently, for the mobile world we need self-protecting systems acting more or less autonomous.

A second drawback applying current network security principles to mobile environments arises from the multi-layered structure of common state-of-the-art network security strategies. They typically define several defence lines trying to protect a network as a whole. As a result, the network structure becomes a part of the configuration of the implemented security measures. The mobile world with its often changing environments renders such an approach impossible. Instead, we postulate the best-possible protection of each single device in accordance with the weakest-link paradigm [2].

In the remainder of this article we describe the challenges a system architecture for self-protecting mobile devices has to meet. The key contributions of the paper are the identification of basic building blocks for such a system as well as the discussion of how to enable collaboration between different devices in order to improve their individual performance. Section II provides an overview of our proposed system architecture. This includes the concept of loose and close cooperation groups for sharing information and resources respectively (section II-D). Thereby we intend to improve the accuracy and efficency of a single device when identifying an attacker. However, this approach raises new security issues that are discussed in section II-E. The main concern is how to protect the communication between previously unknown devices in the absence of central components and how to balance the individual risk of relying on false information. In order to meet these challenges we present a survey of state-of-the-art trust models in section III and analyse their application to our distributed IDS scenario. Section IV gives an overview of the current status of our prototype implementation whereas

related work is summarized in section V. We close with a conclusion of the results already made and a brief summary of open problems as well as the next steps planned.

II. Self-Protecting Mobile Devices

As we have already stated in the previous section a secure mobile device must comply with several more requirements than devices in common wired networks do. These additional requirements result from an ordinary user's lack of security knowledge as well as from the more dynamic structure of mobile environments. Therefore, a secure mobile device must be as far as possible self-protecting, i.e. it must automatically defend itself against malicious attacks. Furthermore, it should use early warning to anticipate such events. This can be realized with the means of autonomic computing [3]. Although in the genesis of autonomic computing research focussed on large-scaled systems like data management or web servers [4] there is also the demand for autonomic personal computing devices. However, autonomic personal computing is a bit harder to achieve as it has to share the goals of personal computing with those of autonomic computing [5].

In the following we summarize our approach for a self-protecting mobile device. For more details we refer to [6]. Figure 1 depicts an architectural model of our system design. It basically consists of three components that are described in detail in the following subsections.

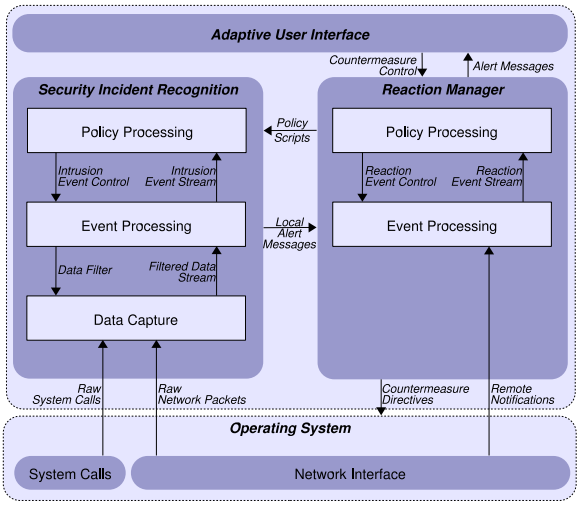

Fig. 1. Basic System Architecture of a Self-Protecting Mobile Device

A. Adaptive User Interface

Complexity of use remains one of the most challenging obstacles of current information technology. Especially security solutions are ill-reputed for being difficult to use [7]. The adaptive user interface on top of our multi-layered architecture tries to minimize the risk of configuration errors as well as to enhance the usability of the whole system even for non-experienced users. Examples for the design of such systems can be found in [8] and [9]. In the context of autonomic computing a major question thereby remains how to decide when security measures can safely be automated and hidden from the user and when an user interaction is indispensable. To get reasonable user input the "questions" asked must be formulated according to his abilities. In other words, the granularity of the informations presented to the user must be adjusted to meet his standard of knowledge.

There already exist several attempts for such systems in practice like anti-virus tools running in background and checking data for an infection when it is accessed. Usually, the user does not notice these tools. Only in case of an incident he is asked if the respective file should be cleaned, wiped out or quarantined for a later detailed inspection. The same applies for personal firewall software on common desktop computers that blocks unwanted network traffic. Unfortunately, as experience shows, these approaches are limited as they reveal the complexity underhood when it comes to the question of configuration: what kind of traffic is unwanted, how can a new virus be detected, how can a a virus be extracted from a file? We believe that a future system must adaptively seek out and leverage the user's (growing) knowledge to overcome these limitations.

B. Security Incident Recognition

To be able to protect itself a device has to detect a security breach first. This is realized by the security incident recognition component (SIR). Basically, the SIR consists of a host-based intrusion detection system observing the mobile device itself and its environment. This is realized by controlling the local network traffic and the local system behaviour. If the SIR detects a (possible) harmful action it informs the reaction manager who decides about further operations.

The SIR's layered architecture results from two important design goals:

- *Separation of mechanism and policy:* A security policy typically depends on the environment it was specified for. Thus, in changing environments like mobile ones a policy must be frequently adopted to new parameters. By clearly seperating the specification of a policy from the mechanisms needed for its observation we achieve both simplicity and flexibility as demanded by the principles of autonomic computing.
- *Efficient operation suitable for low-performance devices:* Common mobile platforms suffer from several resource constraints like processor speed, memory and battery power. By filtering the processed data in several layers we keep the resource requirements as low as possible to assure a smooth functioning of the overall system.

C. Reaction Manager

The reaction manager is quite similar to the SIR. Both have a layered architecture in common resulting from the same design constraints mentioned in the previous section. However, the reaction manager tasks are more high-level. It is the central control instance for the suppression of harmful activities. For this purpose it initiates local countermeasures

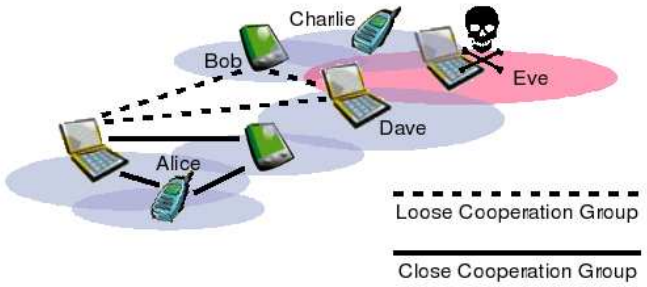

Fig. 2. Improving Attack Detection by Cooperation

by sending corresponding directives to the local operating system, e.g. blocking a specific network address or suppressing a specific system call. Furthermore, it controls the cooperation with allied hosts (see section II-D) by handling incoming warnings from adjacent devices or by broadcasting security notifications to its neighbourhood. It is also responsible for the organization of distributed protection measurements together with trusted appliances.

D. Cooperation Strategies

Our approach is based on the assumption that a mobile network can be protected by securing every single host. However, as each host has only a limited view of the whole network – actually he only knows about the devices within his reach – the distinction between legitimate and harmfull actions becomes a hard problem especially for mobile devices with their resource limitations. To overcome these restrictions we introduce the concept of cooperation groups.

We define a *cooperation group* as a set of several mobile devices working together. A *close cooperation group* is defined as a cooperation group in which the devices are sharing their resources whereas a *loose cooperation group* is a cooperation group in which they only exchange informations. This concept can be compared with a household and a neighbourhood in real life respectively. The participation in a cooperation group is managed by the reaction manager who takes care of joining and leaving a group. Figure 2 shows an example of several mobile users working together in cooperation groups. Alice has bound her personal devices (laptop, mobile phone and PDA) into a close cooperation group so that her mobile can delegate resource consuming tasks to her laptop. In addition she is a member of a loose cooperation group with Bob and Dave, of whom she gets alarmed about the attacker Eve before she comes into her reach.

E. Security Implications

The difference between both types of cooperation groups can be defined by the level of trust the participants are meeting with. In today's computing environments trust is often implemented using cryptographic means like encryption or signature schemes. However, these mechanisms rely on some first-hand knowledge like a public key or a trusted third party. Unfortunately, in the domain described peers are likely not to have such previous knowledge of each other. We can also not rely on the availability of a trusted infrastructure. Together, this makes the usage of cryptographic schemes difficult. In close cooperation groups utilizing mechanisms similar to the pairing process in Bluetooth [10] might be a solution. Bluetooth pairing is used in link establishment between two devices. It involves the same personal identification number (PIN) being entered into each of the devices being linked, resulting in the generation of a symmetric link key for the encryption of the data traffic thereafter. For loose cooperation groups, however, this approach is not practical as the devices are to far away from each other to exchange the PIN. Peer-based mechanisms like the web-of-trust used in PGP [11] might be a solution in this scenario but although they do without central entities they still rely on the fact that most of the peers know each other. Unfortunately, this assumption does not always hold in the described scenario.

As an alternative to traditional cryptography based security mechanisms we propose the utilization of trust to overcome the just mentioned issues.[1] By doing so we intend to answer two basic questions arising from new attack scenarios in which the collaboration itself is the target:

1) How can I be sure that the information received has not been altered if I cannot rely on any infrastructural services or firsthand knowledge?
2) Even if I have protected the integrity of all transmitted data, how can I be sure that my counterpart does not try to cheat?

First thoughts on that will be presented in the following section.

III. Establishing Trusted Partnerships for Secure Collaboration

Although being a rather young area of research in computer science there already exist several publications proposing different concepts and applications for trust and trust management. Artz and Gil did a very good job in discussing nearly one hundred computer science research works about trust which they briefly characterize in four directions: credential-based trust, reputation-based trust, general trust models, and trust in information sources [12]. The main problem with all these research work is that there is no common denominator about the used terminology. In [13] Viljanen describes an attempt to classify thirteen computational trust models by only taking into account the input factors the trust decision is based on. She also presents an ontology of trust to be utilized in digital business.

In the following we first try to precise our view of trust before we identify requirements of our distributed IDS scenario. Finally, we discuss existing trust models in preparation of the design and implementation of an appropriate trust model for securely collaborating intrusion detection systems.

[1] Please note that this statement is not quite precise as there also exist some trust management schemes that use cryptographic mechanisms like for example credentials.

A. What is Trust anyway?

Trust has always been an essential part of our lives. Without trust, there won't be hardly any interaction between people. Consequently, it has been studied by sociologists, psychologists and philosophers for a long time [14]. However, we are more interested in the technical aspects of trust especially, i.e. how automatically determine and process trust. Thus, we have to agree on a definition of trust.

The Oxford Dictionary [15] defines trust as the

> "belief or willingness to believe that one can rely on the goodness, strength, ability, etc of somebody or something."

Thus, trust enables people to cope with the uncertainty caused by the unpredictable behaviour – often called the free will – of the people they are trying to interact with. This uncertainty is also known as the shadow of the future. Jøsang [16] relates trust with malicious behaviour and differentiates between human beings (pationate entities) and systems (rational entities):

> "Trust in a passionate entity is the belief that it will behave without malicious intent ... Trust in a rational entity is the belief that it will resist malicious manipulation by a passionate entity."

A more mathematical definition is presented by Gambetta [17]:

> "trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before he can monitor such action (or independently of his capacity of ever to be able to monitor it) and in a context in which it affects his own action."

A measure to describe this particular level of trust is called a *trust metric*. Based on Jøsang and Gambetta [13] states that trust is neither symmetric, distributive, associative nor transitive. It is also important to record that trust is not only bound to entities but also to a specific action[2]. Thus, trust can be defined as

> "the extent to which one party is willing to participate in a given action with a given partner, considering the risks and incentives involved." [18]

B. Requirements of the Distributed IDS Scenario

The choice of an appropriate trust model for securely collaborating IDS authorities is fundamentally influenced by three basic properties of our mobile setup:

1) We cannot rely on any central entities as they might not be available all the time when travalling around.
2) We must be able to interact with many nodes we have not seen before.
3) We have to avoid excessive user interaction to assure the usability even for unskilled users (see section II-A).

All three properties impede the initialisation of trust relationships.

[2]Imagine your mechanic doing any medical surgery to you.

C. Existing Trust Models

Today, the most common trust models used in distributed system design are either based on credentials or reputation. Credentials are a good way to receive initial trust when interacting with people previously unknown. Unfortunately, when presenting a credential this becomes also subject to trust decisions. A solution for this is to employ a common trusted party to issue and verify credentials. Unfortunately, the availability of such a trusted party is hard to guarantee in a mobile environment.

Reputation-based trust also relies on knowledge. It uses personal experiences to make a trust decision. Additionally one can also consider the experiences of others what we call recommendation. As in credential-based approaches one possibility to trustworthy manage reputation is by using a central, trusted third party. However, the vast majority of current research works follow decentralized approaches. Thus, although there are still some open questions [19] the use of such systems seems to be promising for our distributed IDS scenario.

IV. Prototype Implementation

To validate our approach we have started with a prototype implementation of its core components. Currently, only the Security Incident Recognition component is implemented. In the remainder of this section we describe the status of our implementation and first validation results. For more detailed explanations we direct the reader's attention to [20].

A. Prototype Constraints

We found our prototype implementation on the IEEE 802.11 [21] wireless LAN protocol for three reasons. First, the necessary technical equipment is easy to achieve both in terms of costs and availability. Second, wireless LAN offers a promising playground for laboratory tests as there exist a multitude of possible attack techniques and even ready to use exploits [22]. Third, the IEEE 802.11 defines modes for infrastructural as well as ad-hoc networks.

Furthermore, we utilize the IDS functionality of an already existing IDS for wired networks called Bro [23] by extending its functionality with additional means to monitor and analyze wireless networks. Bro is a Unix-based open source Network Intrusion Detection System designed for passively monitoring high-volume networks in real-time. Its intrusion detection capabilities are based on misuse-detection. The choice for Bro was mainly influenced by two aspects. The first one is based on its layered system architecture. To cope with the huge amounts of traffic it was designed for it reduces the information to be processed by consistently adding new abstraction levels with each layer. Thus, Bro supports both design goals demanded in section II-B.

B. SIR System Architecture

Figure 3 outlines the basic architecture of the SIR component. It follows the layered system architecture of Bro by enhancing each layer to support wireless network traffic.

Please note that there are no modifications necessary to the capturing of network packets as the libpcap library used for this purpose already supports IEEE 802.11. However, we had to extend the other Bro layers to cope with wireless network traffic. In the following we are going to describe these 802.11 addons to the Bro architecture in more detail.

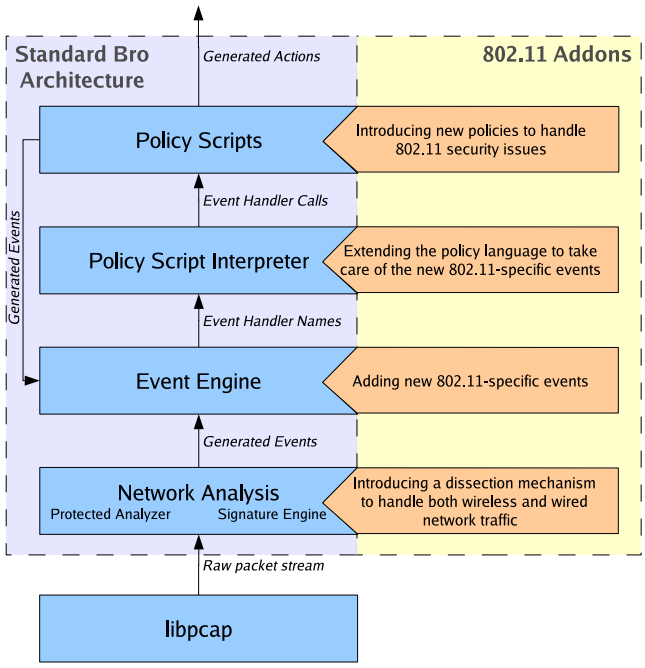

Fig. 3. The SIR Component

Analysing 802.11 frames: First, we added several additional routines to the Network Analyser to extract all important information from IEEE 802.11 header fields and frame body. For example, we determine the link and frame type used. Furthermore, we already do some preliminary processing of data and control frames at this point. Based on the information extracted specific events are generated to trigger the respective policy scripts.

Defining 802.11-specific events: To be able to generate 802.11 specific events we first had to extend Bro's Event Engine by such means. For the moment, we decided to simply map each management frame sub-type to a corresponding event. Thus, the whole traffic analysis can be done at policy level providing us a maximum of flexibility. However, this approach has to be revised most likely when performance becomes an issue.

Enhancing the Bro Policy Layers: Last but not least, we extended the Bro Policy Script Interpreter to cope with the newly introduced 802.11 specific events. In particular, we added built-in support for the 802.11 address type. To detect 802.11 specific attacks we also had to implement several Bro Policy Scripts which will be described in more detail in the following section.

C. First Validation Results and Intended Improvements

To validate our prototype we analysed three common threats to wireless networks: rogue access points, denial-of-service, and man-in-the-middle. For each of these threats we derived a detection strategy which we then translated in a series of specific policy scripts. In the following we proved the effectiveness of our policies during a laboratory test using commonly available hard- and software.

Defining a policy: The process of turning a detection strategy for an attack into a Bro policy basically involves specifying the appropriate event handlers defining how certain events should be interpreted. The challenge, however, is to find a suitable detection strategy in the first place. As an example, we now discuss the derivation of a policy for detecting a 802.11 Denial-of-Service (DoS) attack. One way to conduct a DoS attack described in [24] is to use some form of management frame flooding. For example, an attacker floods the wireless network with spoofed *De-Authentication Frames* that seem to originate from the legitimate access point. Thus, none of the legitimate client stations will be able to stay connected with this access point. On the other hand, an attacker could flood an access point with spoofed *Authentication Frames*, simulating hundreds of individual client stations attempting to authenticate with the access point. Eventually, the access point will be unable to respond to legitimate client requests.

We analysed different kinds of management frame flooding and found out, that they all have one or more of the following characteristics in common:

- exceptional high frequency of certain management frames
- exceptional large number of different source addresses
- destination address set to broadcast address when it should not
- use of invalid source addresses
- unrealistic number of unique network names (SSID) on a single channel

We therefore try to detect flooding attacks by applying basic sanity-checks on the received management traffic. For example, we define several thresholds values for the number of unique source addresses received during a certain period of time (*Authentication Flooding*), the number of unique network names per channel (*Beacon Flooding*) or just the plain frequency of certain management frame types. Additionally, we check for invalid source addresses or the destination field set to multicast addresses where it is not appropriate. Figure 4 shows our basic policy to detect authentication flooding attacks. The actual difficulty with this approach is to find suitable threshold values. Setting them too low would cause too many false alarms while setting them too high could mean that we miss less aggressive attacks.

Testing a policy: We evaluated our prototype in a laboratory test using common hard- and software. Further details about the setup used during the lab test are shown in table I.

Table II summarizes the findings of our laboratory test. Although the tests results are overall promising there is still some space for improvement. For example, the detection of a

```
global flooding_interval : interval = 10 sec;
global flood_counter : count = 0;
global flood_threshold : count = 100;
event wlan_check_flooding() {
  if ( flood_couter > flood_threshold ) {
    print "Flooding-Attack detected!";
  }
  flood_counter = 0;
  schedule flooding_interval {
      wlan_check_flooding()
  };
}
event wlan_auth(c:ieee80211_auth,
                channel:count) {
  flood_counter = flood_counter + 1;
}
event bro_init() {
  schedule flooding_interval {
    wlan_check_flooding()
  };
}
```

Fig. 4. Policy to detect Autentication Flooding

TABLE I
TEST SETUP

Attacker	Debian GNU/Linux 3.1 (Kernel 2.4.29) Netgear WG311 (Atheros chipset, madwifi driver 15.05.2005) Linksys WPC11 (Prism chipset, hostap driver 0.4.1, airjack driver 0.6.6alpha)
Monitor	Debian GNU/Linux 3.1 (Kernel 2.6.12) Netgear WAG511 (Atheros chipset, madwifi driver 15.05.2005) Netgear MA521 (Realtek chipset, driver by Andrea Merello version 0.21)
Victim	Windows XP Professional SP2 Avaya Wireless Silver World Card (Hermes chipset, Windows driver)
Access Point	D-Link DI-624+

man-in-the-middle attack could be improved if all attacked systems, i.e. access point and station, would exchange the results of their analyses. Thus, we are currently working on a communication layer to tie together distinct SIR entities. However, protecting this collaboration from new attack types is still in the early stages as the explanations presented in section III undoubtedly show. Further work is done to analyse additional threats and to make them detectable by adding new policy scripts.

TABLE II
TEST RESULTS

Attack	Tools	Successful	Detected
Rogue Access Point	hostapd,airsnarf	Y	Y
Association-Flooding	void11	N	Y
Authentication-Flooding	void11	N	Y
Deauthentication-Flooding	wlan_jack	Y	Y
Beacon-Flooding	fakeAP	Y	Y
Man-In-The-Middle	monkey_jack	Y	Y
Client MAC-Spoofing	no special tool	Y	N

V. RELATED WORK

Our work was inspired by several publications aiming at similar directions as our approach does. In [25] Ganger and Nagle promote a still informal approach to network security in which each individual device erects its own security perimeter and compare it with the siege warefare analogy that inspired it. The idea of cooperating intrusion detection systems was first presented in 1996 by White, Fisch and Pooch [26]. Since then, there were several similar implementations like EMERALD [27], AAFID [28], Sparta [29] or Indra [30] all neglecting the special needs of mobile environments. The importance of new developments in this direction has been pointed out in [31]. Although the ongoing research in this area is tackling the problems from different perspectives they tacitly rely on the existence of a central network hierarchy and ignore the existence of mobile ad-hoc networks [32], [33]. Most of the works in this area concentrate on routing issues, i.e. how to improve the routing algorithms used to maintain availability in the presence of cheating or misbehaving nodes [1], [34], [35]. Only few works are aware of the unique characteristics of mobile ad hoc networks and their implications to the security design. Like us, [36] considers the lack of a central defence line in mobile networks and demands a complete security solution that integrates proactive and reactive approaches.spanning different devices. However, although addressing the case of collaboration between trusted nodes they do not mention how to implement this trust. [37] discusses the evolution of security threats in wireless local area networks. The authors argue that even though tremendous progress in terms of security has been made there still exist potential threats. Only their type has shifted from outsider attacks to authorized users fighting against each other. In their conclusion the authors demand collaboration as an essential part of a strategy against such attacks. However, they neither mention how to establish such a collaboration nor how to protect it against further threats. Last but not least there exist also some practically oriented works considering infrastructural wireless networks. [32], [38] modify an off the shelf wireless access point in order to detect common wireless attacks and perform active countermeasures against them. [33] even provides an opportunity of physical reaction by localizing the attacker. Snort-Wireless [39] is another attempt to integrate wireless networks within a wired IDS.

VI. CONCLUSION AND PERSPECTIVES

We have presented open issues for the development of self-protecting mobile devices. Compared with conventional autonomic computing systems one has especially to care about the requirements derived from the dynamically changing mobile environment and the user's demands like simplicity of use. On the other hand there are the limitations when adopting security techniques from the wired world. The main contribution of this paper is twofold. First, we proposed a general system architecture to face the mentioned difficulties. Second, we discussed the application of trust as an alternative

to traditional cryptographic schemes to establish a secure collaboration between distinct devices.

We are sure that our approach presents a promising solution although it has not been fully validated yet. This applies especially to the intended use of trustful collaboration in so-called cooperation groups which has – to our best knowledge – not yet been considered by other works in this field. Being merely more than a sketch at the moment we still believe that our solution's potential already becomes visible. We are working hard at the extension of our prototype and hope to present a detailed concept for a trustful collaboration in the near future.

REFERENCES

[1] Y. Zhang, W. Lee, and Y.-A. Huang, "Intrusion Detection Techniques for Mobile Wireless Networks," *Wireless Networks*, vol. 9, no. 5, pp. 545–556, Sept. 2003.

[2] B. Schneier, *Secret and Lies, Digital Security in a Networked World*. Wiley, 2000.

[3] J. O. Kephart and D. M. Chess, "The Vision of Autonomic Computing," *IEEE Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.

[4] A. G. Ganek and T. A. Corbi, "The dawning of the autonomic computing era," *IBM Systems Journal*, vol. 42, no. 1, pp. 5–18, 2003.

[5] D. F. Bantz, C. Bisdikian, D. Challener, J. P. Karidis, S. Mastrianni, A. Mohindra, D. G. Shea, and M. Vanover, "Autonomic personal computing," *IBM Systems Journal*, vol. 42, no. 1, pp. 165–176, 2003.

[6] Stephan Groß, "Selbstschützende mobile Systeme," in *3. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik (Sicherheit 2006)*, ser. Lecture Notes in Informatics (LNI), J. Dittmann, Ed., vol. P-77, Fachbereich Sicherheit der GI. Magdeburg: Gesellschaft für Informatik, Feb 2006, pp. 103–106.

[7] A. Whitten and J. D. Tygar, "Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0," carnegie Mellon University, Pittsburgh, PA. Retrieved April, 12 2005 from http://www.gaudior.net/alma/johnny.pdf.

[8] G. Wolf and A. Pfitzmann, "Properties of Protection Goals and their Integration into a User Interface," *Computer Networks, Special Issue on Electronic Commerce*, vol. 32, 2000.

[9] A. Whitten, "Making Security Usable," Ph.D. dissertation, Carnegie Mellon University, May 2004, cMU-CS-04-135.

[10] B. S. I. Group, "Bluetooth 1.1 Core specification," Retrieved April, 12 2005 from http://www.bluetooth.org/spec.

[11] S. Garfinkel, *PGP. Pretty Good Privacy. Encryption for Everyone.* O'Reilly, 1994.

[12] D. Artz and Y. Gil, "A Survey of Trust in Computer Science and the Semantic Web," 2006, submitted for publication. [Online]. Available: http://www.isi.edu/~dono/pdf/artz06survey.pdf

[13] L. Viljanen, "Towards an Ontology of Trust." in *TrustBus*, ser. Lecture Notes in Computer Science, S. K. Katsikas, J. Lopez, and G. Pernul, Eds., vol. 3592. Springer, 2005, pp. 175–184.

[14] S. Marsh, "Formalising Trust as a Computational Concept," Ph.D. dissertation, University of Stirling, UK, 1994.

[15] A. P. Cowie, Ed., *Oxford Advanced Learner's Dictionary*. Oxford University Press, 1989.

[16] A. Jøsang, "The right type of trust for distributed systems," in *Proceedings of the 1996 New Security Paradimgs Workshop*, C. Meadows, Ed. ACM-Press, 1996.

[17] D. Gambetta, *Can We Trust Trust?* Department of Sociology, University of Oxford, 2000, ch. 13, pp. 213–237, electronic edition. [Online]. Available: http://www.sociology.ox.ac.uk/papers/trustbook.html

[18] S. Ruohomaa and L. Kutvonen, "Trust Management Survey," in *iTrust 2005*, ser. LNCS, P. H. et al., Ed., no. 3477. Springer-Verlag, 2005, pp. 77–92.

[19] S. Moloney and P. Ginzboorg, "Security for Interactions in Pervasive Networks: Applicability of Recommendation Systems," in *ESACS 2004*, ser. LNCS, C. C. et al., Ed., no. 3313. Berlin Heidelberg: Springer-Verlag, 2005, pp. 95–106.

[20] René Neumerkel and Stephan Groß, "A Sophisticated Solution for Revealing Attacks on Wireless LAN," in *Proceedings of the 3rd International Conference on Trust, Privacy, and Security in Digital Business (TrustBus '06)*, ser. LNCS. Krakow, Poland: Springer Verlag, Sep 2006, accepted for publication.

[21] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, ANSI/IEEE Std 802.11," ANSI/IEEE, Tech. Rep., 1999. [Online]. Available: http://standards.ieee.org/getieee802/

[22] A. Vladimirov, K. V. Gavrilenko, and A. A. Mikhailovsky, *WI-FOO. The Secrets of Wireless Hacking*. Addison-Wesley Professional, 2004.

[23] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-Time," *Computer Networks*, vol. 31, no. 23–24, pp. 2435–2463, Dec. 1999.

[24] J. Bellardo and S. Savage, "802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions," in *Proceedings of the 12th USENIX Security Symposium*, Washington D.C., Aug 2003, pp. 15–28. [Online]. Available: http://www.usenix.org/events/sec03/tech/full_papers/bellardo/bellardo.pdf

[25] G. R. Ganger and D. F. Nagle, "Better Security via Smarter Devices," in *Proceedings of the 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII)*. Elmau, Germany: IEEE, May 2001.

[26] G. B. White, E. A. Fisch, and U. W. Pooch, "Cooperating Security Managers: A Peer-Based Intrusion Detection System," *IEEE Network*, vol. 10, no. 1, pp. 20–23, January/February 1996.

[27] P. A. Porras and P. G. Neumann, "Emerald: Event monitoring enabling responses to anomalous live disturbances," in *20th National Information Systems Security Conference*, Oct. 1997, pp. 353–365.

[28] E. H. Spafford and D. Zamboni, "Intrusion detection using autonomous agents4," *Computer Networks*, no. 34, pp. 547–570, 2000.

[29] C. Krügel, T. Toth, and E. Kirda, "Sparta – a mobile agent based intrusion detection system," in *IFIP Conference on Network Security*. Belgium: Kluwer Academic Publishers, 2001.

[30] R. Janakiraman, M. Waldvogel, and Q. Zhang, "Indra: A peer-to-peer approach to network intrusion detection and prevention," in *12th IEEE International Workshop on Enabling Technologies (WETICE 2003), Infrastructure for Collaborative Enterprises*, Linz, Austria, June 2003.

[31] L. Buttyán and J.-P. Hubaux, "Report on a Working Session on Security in Wireless Ad Hoc Networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 1, pp. 74–94, Jan. 2003.

[32] Y.-X. Lim, T. Schmoyer, J. Levine, and H. L. Owen, "Wireless Intrusion Detection and Response," in *Proceedings of the 2003 IEEE Workshop on Information Assurance*, United States Military Academy, West Point, NY, USA, Jun 2003.

[33] J. W. Branch, N. L. P. Jr., L. V. Doorn, and D. Safford, "Autonomic 802.11 Wireless LAN Security Auditing," *IEEE Security & Privacy*, vol. 2, no. 3, pp. 56–65, May 2004.

[34] R. S. Puttini, L. Mé, and R. T. de Sousa Jr., "Preventive and Corrective Protection for Mobile Ad Hoc Network Routing Protocols," in *Wireless On-Demand Network Systems*, ser. LNCS, vol. 2928. Springer-Verlag Heideberg, Dec. 2003, pp. 213–226.

[35] R. S. Puttini, J.-M. Percher, L. Mé, O. Camp, R. de Sousa Jr., C. J. B. Abbas, and L. J. García-Villalba, "A Modular Architecture for Distributed IDS in MANET," in *Computational Science and Its Applications – ICCSA 2003*, ser. LNCS, V. Kumar, M. L. Gavrilova, C. J. K. Tan, and P. L'Ecuyer, Eds., vol. 2669. Montreal, Canada: Springer-Verlag Heidelberg, Aug. 2003, pp. 91–113.

[36] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, "Security in Mobile Ad Hoc Networks: Challenges and Solutions," *IEEE Wireless Communications*, vol. 11, no. 1, pp. 38–47, Feb 2004. [Online]. Available: http://repositories.cdlib.org/cgi/viewcontent.cgi?article=2292&context=postprints

[37] R. A. Beyah, C. L. Corbett, and J. A. Copeland, "The Case for Collaborative Distributed Wireless Intrusion Detection Systems," in *Proceedings of the IEEE International Conference on Granular Computing (GrC)*, 2006. [Online]. Available: http://www.cs.gsu.edu/rbeyah/HotspotWorm_cr.pdf

[38] T. R. Schmoyer, Y. X. Lim, and H. L. Owen, "Wireless Intrusion Detection and Response. A case study using the classic man-in-the-middle attack," in *Proceedings of the IEEE Wireless Communications and Networks Conference*, Atlanta, Georgia, USA, Mar 2004.

[39] "Snort-Wireless Project Homepage." [Online]. Available: http://snort-wireless.org/