User-Based Attestation for Trustworthy Visual Sensor Networks

Thomas Winkler
Pervasive Computing Group
Institute of Networked and Embedded Systems
Klagenfurt University
Lakeside Park B02b
9020 Klagenfurt, Austria
Email: thomas.winkler@uni-klu.ac.at

Bernhard Rinner
Pervasive Computing Group
Institute of Networked and Embedded Systems
Klagenfurt University
Lakeside Park B02b
9020 Klagenfurt, Austria
Email: bernhard.rinner@uni-klu.ac.at

***Abstract*—Camera networks are used for a variety of applications including surveillance, traffic monitoring or elderly care. The shift from analog towards fully digitized systems has considerably increased their capabilities. With large-scale deployments of smart cameras and visual sensor networks, public awareness of privacy issues is increasing. Researchers are addressing these concerns by introducing privacy preserving technologies like content scrambling and encryption. Today's systems however do not provide mechanisms that allow monitored people to verify that a camera system is behaving as advertised by its operators. In this work, we propose to use Trusted Computing to enhance the security of camera systems and, by enabling user-based attestation, give users a simple and intuitive way to check the trustworthiness of cameras.**

I. Introduction and Motivation

Video cameras have become a part of our daily life. In London for example, an average citizen is caught on CCTV cameras 300 times a day [1]. Many of today's camera systems are fully digitized and do onboard image processing and analysis. In Visual Sensor Networks [2], large numbers of small, cheap, and wirelessly networked cameras are deployed. Inter-camera communication allows to, e.g., track persons over large distances [3]. At the same time, public awareness and concerns about privacy issues in video surveillance are growing. Researchers have proposed solutions that address these issues by, e.g., scrambling or encrypting privacy sensitive images regions [4], [5], [6].

In future systems, these privacy preserving techniques might be adopted by manufacturers and operators to increase acceptance of camera systems. Monitored people however would still have to blindly trust that these mechanisms are enabled and that the cameras behave as advertised. In this work we present an approach that empowers people to actually verify that cameras in their environment are trustworthy. Our user-based attestation concept makes use of Trusted Computing (TC) and proposes to equip smart cameras with a dedicated hardware security chip known as Trusted Platform Module (TPM). With a handheld device, users can establish an authenticated channel to a camera based on visual communication. This channel is used to attest the state and trustworthiness of the camera device with the help of the TPM. The attestation result is not a simple, binary trust decisions. Properties assigned to the camera's software state allow the user to learn if, e.g., the camera streams video or not, or if sensitive image regions are encrypted. Our proposal assumes cooperation of camera network operators. We believe that is is reasonable, as our approach would raise public acceptance of camera networks.

This paper is structured as follows: In section II we summarize related work in the areas of privacy protection techniques for camera systems and user-based attestation. After discussing the fundamentals of TC in section III, section IV presents our system architecture. It is followed by details of our approach for user-based attestation in section V. Our prototype implementation and evaluation results are discussed in section VI. Section VII highlights open issues together with future work and finally concludes the paper.

II. Related Work

In previous work [7] we applied TC to address security requirements of camera operators. We now shift our attention to the needs of the persons monitored by the cameras. Privacy is one of the most critical issue to users. The following sections summarize work related to privacy in video surveillance and concepts for user-based attestation.

A. Privacy in Camera Networks

Moncrieff et al. [3] present an overview of the understanding of the term privacy in the context of ubiquitous computing. Even though the notion of privacy is highly subjective and strongly varies across cultures, there are a number of aspects agreed upon by many researchers. These include that users should have control over what data is captured, how it is processed, shared and used. The authors argue that the move from traditional CCTV systems towards fully digitized systems has a strong impact on user privacy. Digital video footage is easily storable, can be indexed for searching and can easily be retrieved. Moreover, networks of cameras allow to cover large areas and to track persons from camera to camera. Slowly but steadily, public awareness of the involved privacy issues is growing. The authors propose

to address these concerns by applying dynamic data hiding techniques. While during normal operation privacy sensitive data is removed, in case of, e.g., an alarm the system dynamically is adapted to reveal more information. This way, the system remains usable for the intended purpose but protects privacy during normal operation.

Cavallaro [1] specifically highlights the threat of operator misuse. He proposes to follow an approach where operating staff is only provided with a stream of abstract metadata while a separate stream containing personal video data is only made available to law enforcement authorities.

Serpanos et al. [8] present an extensive overview of security and privacy related issues in smart camera networks. They discuss the need for confidentiality, integrity and freshness of data transmitted between nodes. In cases where images are sent, privacy of observed persons is a critical issue as it not only involves protection of sensitive information against external attackers but also against legitimate system operators. To achieve this goal, relevant parts of the images need to be recognized and appropriately encrypted.

Senior et al. [9] discuss the meaning of privacy in video surveillance and conclude that there is no general notion of privacy but what is acceptable depends on the individual person and cultural attitudes. They discuss critical aspects of a surveillance system including what data is available and in what form (e.g., raw images vs. metadata), who has access to data and in what form (e.g., plain vs. encrypted) and how long it is stored. Finally, they propose a system that preserves user privacy by pre-processing videos on the camera and a layered approach for granting access to the different types of information produced by the camera.

Data hiding techniques that allow to mask sensitive image regions on a smart camera system have been proposed by a number or researchers. With PrivacyCam [5] Chattopadhyay et al. present a system based on a Blackfin DSP which identifies regions of interest based on a background subtraction model. Resulting regions are encrypted using an AES session key. Baaziz et al. [4] also perform motion detection for scrambling. To additionally ensure data integrity, they embedded a watermark into the image. This allows to detect manipulation of image data and limited reconstruction of manipulated image regions due to introduced redundancy. In similar work, Dufaux et al. [6] do not rely on cryptographic primitives for content protection but propose to integrate content scrambling into MPEG-4 and MJPEG encoding processes. Systems that support different levels of object masking, e.g., fully blanking sensitive regions, revealing only silhouettes or replacing detected persons by a label have been demonstrated in [10] and [7].

B. User Based Attestation

The primary goal of user based attestation is to provide a mechanism where users can verify the state of a platform in an ad-hoc manner. A major problem highlighted by Parno [11] is the absence of a reliable way to establish the identity of a TPM inside a computer. As a consequence, a malicious machine could forward TPM related requests of a user to another TPM-enabled, unmodified machine which then would provide valid response messages. This type of attack is called a *cuckoo attack*. The author argues that the establishment of the TPM identity hence is a fundamental precondition for reliably attesting the software state of a platform. In conclusion, the work suggests to add a special-purpose hardware interface that allows an external device to directly communicate with a TPM.

For the purpose of trustworthy kiosk computing, Toegl [12] extends this idea and proposes the integration of an Near Field Communication (NFC) interface into the TPM. Via the NFC interface, a user with a trusted, NFC enabled handheld device can set a nonce into a dedicated register of the TPM. This nonce is then included in the subsequent TPM_Quote operation. The establishment of the nonce requires the user to bring the handheld into close proximity (a few centimeters) of the TPM. This ensures that the attestation response actually comes from the intended machine. As the NFC based establishment of the nonce bypasses the software stack of the host machine, malicious software on the host can not manipulate the attestation process.

With Seeing-Is-Believing (SIB) [13], McCune et al. take a different approach using visual communication to establish an authentic communication channel between mobile phones. Visual communication has the advantages that it is intuitive to use and attacks on the communication are easily spotted. In this procedure, called demonstrative identification, a 2D barcode containing a key is displayed by one smartphone which then is captured using the camera of the second phone. Subsequently also performing this procedure in the opposite direction, allows to establish a mutually authenticated communication channel. In cases where one of the devices does not have a display, the authors propose to attach a sticker with the printed barcode to the system. This approach is also proposed by Garris et al. [14] in their work targeted towards the realization of trustworthy and personalized computing environments on public kiosks. However, as discussed in [11], [12] this approach is problematic because stickers are easily modified or replaced and hence can not help to reliably prevent cuckoo attacks. Bangerter et al. [15] also use the visual channel together with a dedicated, proprietary security token to attest the state of a system. Using this device, a logical and secure channel between the token and an attestation server is established. Messages from the server are sent to the token by flickering the screen of the attested system. The message encoded in this flickering is captured by the token's camera.

Other researches pursue similar ideas but use different communication techniques to establish a local, authentic channel. With Loud and Clear, Goodrich et al. [16] propose a system that uses audio communication for device pairing.

In this system, a human user is required to compare english phrases which encode authentication data played by the involved devices. The authors argue that one advantage of the system is that it can operate over larger distances than, e.g., visual solutions. This however also makes the system more vulnerable to cuckoo attacks as identification of the talking device might not be as intuitive as with visual approaches.

III. Trusted Computing Preliminaries

TC is an industry initiative headed by the Trusted Computing Group (TCG). The main output of the group is a set of specifications for a hardware chip – the Trusted Platform Module (TPM) [17] – and software infrastructure like the TCG Software Stack (TSS) [18]. The TPM typically is implemented as a microcontroller (execution engine) with accelerators for RSA and SHA1. Additionally, the TPM provides a random number generator and limited amount of volatile and non-volatile memory. With an Opt-In process, users can choose if they want to make use of the TPM.

RSA keys can be generated for different purposes like encryption or signing. Upon creation, keys can be declared migratable or not. While migratable keys can be transferred to a different TPM, non-migratable keys can not. Regardless of key type and migratability, a private TPM key can never be extracted from the chip as plaintext but only in encrypted form. By definition, every key must have a parent key that is used to encrypt the key when it has to be swapped out of the TPM due to limited internal memory. At the top of this key hierarchy is the Storage Root Key (SRK) which never leaves the TPM. TC defines three roots of trust:

- **Root of Trust for Measurement (RTM).** In TC, measuring is the process of computing the SHA1 hash of an application binary before it is executed. Typically starting from an immutable part of the BIOS, a chain of trust is established where each component in the chain is measured before control is passed to it. The measurements are stored inside the TPM in memory regions called Platform Configuration Registers (PCRs). As available memory in the TPM is limited, a special operation called TPM_Extend is used to write to PCRs:
$$PCR[i] \leftarrow SHA1(PCR[i]||measurement).$$
TPM_Extend computes the hash of the current PCR value concatenated with the new measurement. This accumulated value is written back into the PCR.
- **Root of Trust for Reporting (RTR).** Reporting of the platform state is called attestation and is done with the TPM_Quote command. As part of that, PCR values get signed inside the TPM using a key unique to that TPM. In theory, this key could be the Endorsement Key (EK) which is inserted into the TPM upon manufacturing. For privacy reasons however, not directly the EK but alias keys are used. They are called Attestation Identity Keys (AIKs) and are generated with the help of an external trusted third party.
- **Root of Trust for Storage (RTS).** The RTS allows to use the TPM to securely store data. Binding of data refers to encrypting data with a TPM key and hence guaranteeing that this data only is accessible by this specific TPM instance. Sealing of data allows to specify a set of PCR values the data is associated with. As with binding, the unsealing can only be done by the specific TPM instance that holds the private sealing key. Additionally, the plaintext is only released if the current PCR values match those specified upon sealing.

IV. System Architecture

One focus of this work is to provide users a mechanism that allows them to query the state of a camera installed in their environment. To facilitate secure reporting of the camera state, we follow the TC remote attestation approach.

A. Targeted User Experience

Our goal is the design of an intuitive mechanism that enables users to (1) select the camera they are interested in, (2) perform remote attestation of the camera and, based on the attestation result, (3) find out what the applications on the camera are doing.

Users who want to attest a camera are equipped with a trusted handheld device. To select a specific camera, users walk up to the camera and point the handheld with its display towards the camera. The handheld displays a 2D barcode that encodes all information relevant for starting the attestation procedure as detailed in section V. Once attestation is complete, the outcome is displayed on the users handheld. This should not only be a high level trust decision, but also provide the user with additional information about the image processing and analysis applications running on the camera together with their properties.

B. System Components

The overall system design is based on our previous work described in [7]. As shown in figure 1, each camera is equipped with a TPM called TPM_C. Cameras are controlled and managed from a back office with dedicated computing infrastructure. The control station at the back office is equipped with a TPM called TPM_S used for secure communication between cameras and the control station [7]. In this work we assume that the cameras are protected from physical manipulation, e.g., via appropriate enclosure. In addition, we introduce an a priori trusted handheld device which a user employs for attesting cameras. This handheld is at least equipped with a display, a wireless communication interface and buttons or a touchscreen for interaction.

A trusted third party (TTP) is responsible for (1) issuing AIK certificates during camera setup and (2) acting as verification instance during attestation. This verification includes

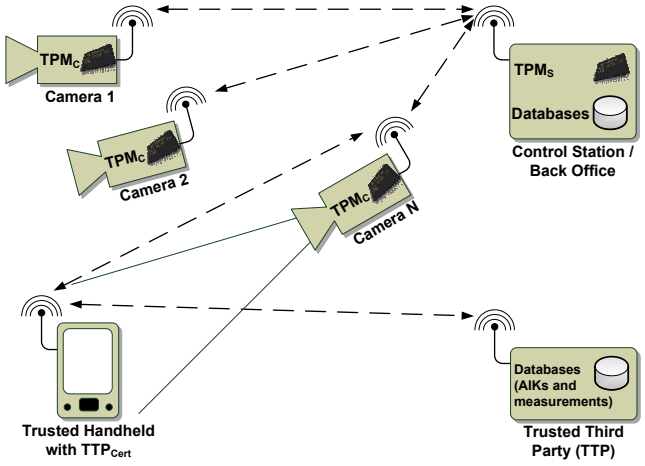

Figure 1. System Architecture Overview.

validation of quote data generated by the camera as well as translating this result into system properties comprehensible to the user. Issuing AIK certificates and checking quote results does not necessarily have to be performed by the same entity. We however have chosen this approach to simplify the following descriptions. To protect communication with the TTP, the trusted handheld is pre-loaded with the TTP's public key certificate (TTP_{Cert}).

C. System Setup

Before a camera can be deployed, a number of setup steps have to be performed. It is assumed that cameras are under full control of the operating personnel during setup. As far as the TPM and user-based attestation are concerned, the setup steps are:

- **TPM Ownership.** Via the TPM_TakeOwnership command, a randomly generated, unique owner password is set for TPM_C. For maintenance operations, the owner password is stored in the control station database. As part of taking TPM ownership, also the SRK is generated.
- **Identity Key Creation.** For user-based attestation, one single AIK is generated. The intention of an AIK is to act as an alias for the TPM's unique EK during platform attestation. For privacy protection, a user has the freedom to generate and use any number of AIKs. In the context of an embedded camera system however, there are no users "on the system" as, e.g., on a desktop PC. Consequently, we only create a single AIK as anonymity in this context is not an issue. This must however not be mistaken with protecting privacy of monitored people. As part of the AIK creation, an AIK certificate is issued by the TTP which acts as a PrivacyCA [19] for the camera network. The AIK certificate vouches for the fact that this AIK actually is an identity key protected by a genuine TPM belonging to one of the cameras of the network.
- **Signature Key Creation.** For signing images sent from the camera to the handheld, a non-migratable singing key K_{SIG} is created with K_{SRK} as its parent. Being non-migratable ensures that the private key is protected by the cameras TPM_C and can only be used inside this specific TPM_C. This provides assurance that data signed with this particular key really originates from this specific camera.

When a camera boots, a chain of trust has to be established that ensures that every relevant component gets measured before it is executed. For our camera system [7], we proposed an approach where we use a static root of trust for measurement (e.g., implemented as ROM) that measures the bootloader which in turn measures the OS kernel and its parameters. Finally, the basic firmware image gets measured which then is mounted read-only. Additionally, each computer vision application executed by the camera gets measured into the PCRs. This approach allows to keep the number of measurements small while being able to make an assertion about the system state and its properties.

As it is the goal to report properties of an attested camera to the user, a procedure is required that translates measurements of vision applications and their configuration into properties. These properties could be, e.g., if the system streams video, if sensitive image regions are encrypted, or which statistics are gathered by the system. In our concept, the TTP shown in figure 1 is responsible for translating the measurements into properties. For that purpose, the camera manufacturer or operator has to submit the applications together with the source code to this TTP for review. Based on the properties reported by the TTP, users gain insight into the behavior of the camera.

V. Visual User-Based Attestation

A main challenge of user-based attestation is the proper selection and authentication of the intended camera. To be feasible to average users, this process needs to be intuitive and largely automated. At the same time, it must be ensured that cuckoo attacks are properly prevented. Typically cameras are not mounted in places easily reachable by users. Consequently, a dedicated hardware interface to the TPM is not an option. Similar considerations hold true for NFC communication. A more natural choice for a camera system is the visual channel. Existing approaches like SIB would allow us to authenticate the handheld device via the camera. For the other direction where the handheld authenticates the camera, the camera would need a display or a barcode sticker attached to it. As there is little use for a display on a camera and stickers are easily manipulated [14], [12], we now present an alternative approach that does not rely on barcode stickers or displays on the camera.

Table 2 shows our user-based attestation protocol. It consists of two phases separated by two horizontal, black

	Verifier (User with Trusted Handheld)		Smart Camera
1	generate 2D barcode with quote request, nonce N_1, PCR_{List} and the handheld IP address		
2		show barcode to camera $\longrightarrow$	
3			from 2D barcode extract nonce N_1, PCR_{List} and IP address perform TPM_Quote: $Quote_{Res} = TPM\_Quote_{K_{AIK}}(N_1, PCR_{List})$
4		$\longleftarrow$ via wireless connection to IP send ($Quote_{Res}$, $Cert_{AIK}$, PCR_{Vals}, PCR_{Log})	
5	validate $Cert_{AIK}$ (contact TTP of camera network) $\rightarrow$ assurance that quote comes from a TPM equipped camera belonging to the network send $Quote_{Res}$, PCR_{Vals} and PCR_{Log} to TTP for validation. TTP returns signed trust decision and list of system properties $\rightarrow$ assurance that camera is in trustworthy state		
6	generate 2D barcode containing IP, nonce N_2 and $GrabAndSignImg$ request		
7		show barcode to camera $\longrightarrow$	
8			perform $GrabAndSignImg$ consisting of reading image img from sensor and signing img: $Sig_{img} = TPM\_Sign_{K_{SIG}}(img)$ certify signing key: $cert = TPM\_CertifyKey_{K_{AIK}}(K_{SIG_{pub}, keyInfo})$
9		$\longleftarrow$ via wireless connection to IP send (img, Sig_{img}, $K_{SIG_{pub}}$, $cert$)	
10	$verify_{K_{SIG_{pub}}}(img, Sig_{img})$ $verify_{K_{AIK_{pub}}}(cert, K_{SIG_{pub}})$ extract nonce N_2' from img and compare it to N_2 $\rightarrow$ assurance that img is from the same camera as the quote (i.e. the one in front of the user)		
11	present the attestation result to the user		

Figure 2. The user-based attestation protocol flow.

lines. The first phase starts with the generation of a 2D barcode on the user's handheld device in step 1. This barcode contains a TPM_Quote request together with a randomly generated nonce N_1, the list of the PCRs to be quoted and the IP address of the handheld's wireless interface. Next, the user presents the barcode to the camera to be attested by pointing the handheld with the displayed barcode towards the camera. In step 3, the camera captures an image and extracts N_1, the list of PCRs and the IP address from the barcode. It then performs the TPM_Quote command using K_{AIK}:

$$Quote_{Res} \leftarrow TPM\_Quote_{K_{AIK}}(N_1, PCR_{List}).$$

Subsequently, a wireless connection is established to the IP of the user's handheld and in step 4, the signed quote result $Quote_{Res}$, the PCR measurement log PCR_{Log} and the AIK certificate $Cert_{AIK}$ are sent back to the handheld.

Using this data, the handheld has to perform the following two steps: (a) With the help of the external TTP, it has to be verified that $Cert_{AIK}$ was issued for an AIK protected by a TPM that is part of a camera of the network. Furthermore, it has to be checked that the certificate was not revoked. (b) The signature of the quote result $Quote_{Res}$ has to be verified and the content of the quote blob has to be examined. This includes comparing N_1 as well as evaluating the provided PCR values together with the PCR measurement log PCR_{Log}. To offload work from the handheld, we submit the quote blob and the PCR log to the TTP which evaluates the blob in conjunction with the log. The individual PCR values are compared to the hashes of the firmware and the applications that were submitted for review by the camera manufacturer or operator. As a result, the TTP returns a signed trust decision and a signed set of properties that describes the behavior of the executed applications.

If all checks succeeded, the user now has assurance that (a) the quote came from *a* camera that belongs to the network and (b) the camera is in a trustworthy state. The user however does not yet have assurance that the quote actually came from the camera the 2D barcode was presented to. This specific camera might have been subverted by an

attacker. Instead of performing a local quote revealing this fact, malicious software on the camera could grab an image, extract N_1, PCR_{List} and IP and forward this data to an unmodified camera. This camera then responds with a valid quote result. This would lead the user to believe that the camera in front of the user is in the reported, trustworthy state while it actually is running malicious software.

To eliminate this attack pattern, we introduce the second phase of the attestation protocol represented by the lower part of table 2. This phase starts with step 6 where a new 2D barcode is generated by the user's handheld that includes a $GrabAndSignImg$ request, a new nonce N_2 and the IP address of the verifier's handheld. This barcode is presented to the same camera as the first barcode. As part of the $GrabAndSignImg$ function in step 8, the camera reads an image from the sensor. This image, showing nonce N_2, is signed with the non-migratable TPM signing key K_{SIG}:

$$Sig_{img} \leftarrow TPM\_Sign_{K_{SIG}}(img).$$

Next, K_{SIG} is certified using K_{AIK}:

$$cert \leftarrow TPM\_CertifyKey_{K_{AIK}}(K_{SIG_{pub}}, keyInfo).$$

The certificate $cert$ consists of the signed hash of the public signing key $K_{SIG_{pub}}$ and the TPM_CERTIFY_INFO2 structure that contains information about the key (e.g., non-migratable etc.). In step 9, the original image, the image signature Sig_{img}, the public signature key $K_{SIG_{pub}}$ and the certificate $cert$ are sent back to the handheld. In step 10, the application on the handheld has to perform the following three verification steps: (a) The image signature Sig_{img} has to be verified. (b) The certificate $cert$ of K_{SIG} must be verified using the public AIK from $Cert_{AIK}$ which was also used for quote validation in step 5. This ensures that the quote and the signed image come from the same camera. (c) From the barcode of the signed image nonce N_2' has to be extracted and compared with N_2 to ensure freshness.

If these three steps succeeded, the user knows that the quote in step 3 and the image signature in step 8 were performed by the same TPM and hence come from the same camera. Our concept assumes that one property of the trustworthy state reported in step 5 is that the camera does not offer a remotely accessible signing function where the TPM signs arbitrary, externally provided data. The only available signing function, triggered via the visual channel, is $GrabAndSignImg$. Consequently, the camera in trustworthy state would not sign an image forwarded to it, e.g., via wireless communication as part of a cuckoo attack. Assuming that N_2 and N_2' extracted from the signed image are identical, it is ensured that the second barcode was seen by the trustworthy camera and that this camera is the one in front of the user. In step 11, the attestation result and the properties reported by the TTP are presented to the user.

Finally it must be noted, that our proposed approach does not limit a trustworthy camera to perform a TPM_Quote only when requested visually. The motivation for this is that we want to ensure that operators are able to check the state of a camera from remote as we described in [7].

VI. Prototype Implementation

Figure 3 shows a smart camera prototype we developed for VSN applications. It is equipped with a dual-core OMAP3530 CPU with an ARM Cortex A8 (480 MHz) and a DSP (430 MHz). It provides 128 MB RAM and 256 MB NAND flash. Via USB, we connect a Logitech QuickCam S5500 (color, VGA), an RA-Link RA-2571 802.11b/g WiFi adapter and a SunSPOT mote for 802.15.4 wireless networking. As operating system we use Debian GNU/Linux compiled for ARM with an OMAP specific kernel.

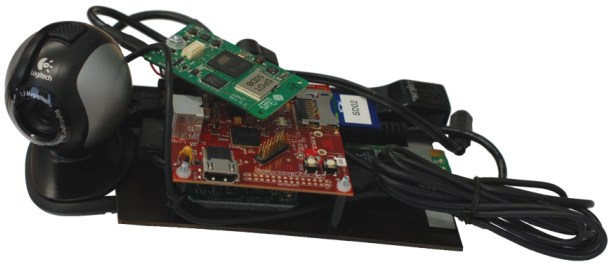

Figure 3. Smart camera research prototype.

As our prototype is not equipped with a hardware TPM, we rely on a TPM emulator [20] for application level TC integration. To establish the chain of trust, as shown in figure 4, we assume that the system incorporates a static RTM implemented as ROM. This RTM initially measures the uBoot bootloader which in turn measures the Linux kernel and its parameters. To keep the number of measurements small, we suggest to next measure the entire root file-system before it is mounted read-only. The file-system image includes a central application called *NodeManager* that is responsible for camera management. It is the only entity that starts and monitors the actual computer vision processing blocks. To provide additional information on running image processing tasks, the *NodeManager* measures every started processing block into PCRs as shown in figure 4. This way, the verifier can learn which tasks are executed without being overwhelmed by an extensive set of PCR values.

As trusted handheld, we use a Nokia N810 internet tablet equipped with an OMAP2420 CPU, a 4.1 inch touchscreen, a WiFi interface and Maemo Linux. For barcode generation and reading we use the dmtx[1] data matrix library.

The 2D barcodes generated on the handheld contain the request id (1 byte; Quote or GrabAndSignImage), the nonce N_x (20 bytes) and the handheld IP address (4 bytes). Currently we omit the PCR list and quote all PCRs. The total 25 bytes are encoded in a data matrix with a symbol size of 22x22. Barcode generation takes about 2.5 ms. The

[1] libdmtx: http://www.libdmtx.org/ (visited Nov. 2009)

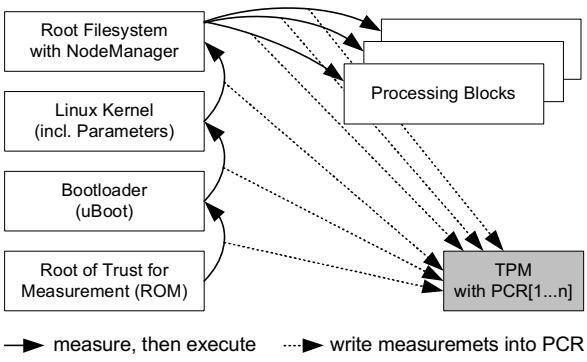

Figure 4. The smart camera's chain of trust.

displayed barcode, as shown in figure 5, has a size of 5x5 cm and is pointed towards the camera. To avoid system overload, we do not search every captured frame for a barcode but analyze one frame every 5 s. Finding and extracting the barcode data takes around 280 ms. Practical experiments with our setup have shown that barcode detection works satisfying for distances up to 40 cm. While this might seem low, it is adequate for our application. In a practical setup, a telescopic extender can be used to bring the handheld sufficiently close to the camera. Moreover, the short distance eliminates the risk that the barcode is also captured by an adjacent camera not intended by the user. In additional tests with a 12 inch tablet PC, we achieved distances of more than 130 cm. Performance of visual tag systems continues to evolve and novel systems like Bokode [21] are reported to work for distances of more than 4 m.

Figure 5. A visual quote request.

For TPM access, we use the TrouSerS [2] software stack. In table I we give the runtimes for the relevant TPM commands. We measured the execution times for the TPM emulator on the camera as well as the runtimes for Infineon and Atmel 1.2 TPMs. While the former is among the fastest available hardware TPMs [7], the later is the only one available with an interface suitable for embedded systems (I2C). In total, the TPM commands TPM_Quote, TPM_Sign, TPM_CertifyKey, multiple TPM_OIAP calls for TPM command authorization together with TSS overhead takes 340 ms with the emulator. With the Infineon chip, this accumulated runtime increases to 1240 ms and with the Atmel TPM it goes up to 2690 ms. It is worth mentioning that even if runtimes of the hardware TPMs are higher, they have less impact on overall performance as the commands run in parallel to the vision applications on the main CPU.

Total runtime of our visual attestation prototype is made up of the runtimes for the barcode operations, the TPM commands plus additional overhead for communication. Using the TPM emulator, this accumulated runtime is 1 s. With an Infineon TPM this would go up to 1.8 s and with the Atmel chip to 3.2 s. These total runtimes are lower bounds as they do not include TTP interaction. Moreover, they do not reflect delays introduced by doing barcode detection only at a predefined interval.

Operation	Runtime
TPM_OIAP	2.9 ms / 28.6 ms / 44 ms [3]
TPM_Quote	78.6 ms / 353.5 ms / 827.1 ms [3]
TPM_Sign	77.5 ms / 340.0 ms / 792.6 ms [3]
TPM_CertifyKey	84.9 ms / 366.4 ms / 845.6 ms [3]
TSS overhead / command	30 ms
barcode reading (camera)	∼280 ms
barcode creation (n810)	2.5 ms
barcode reading (n810)	∼330 ms
communication overhead	8 ms
total (lower bound):	∼960 ms / ∼1860 ms / ∼3310 ms [3]

Table I
VISUAL ATTESTATION RUNTIME ANALYSIS.

To complete our prototype, we implemented a minimal TTP on a laptop. It acts as PrivacyCA and performs the mapping of PCR measurements to properties. In the prototype, these properties are limited to (1) *camera streams full video*, (2) *camera streams video with encrypted motion regions* and (3) *camera does not stream video*. After the attestation of the camera is complete, one of these properties is displayed on the user's handheld.

VII. Conclusions and Future Work

In this work we proposed an intuitive mechanism that enables users to check the trustworthiness of smart cameras. Even tough the system is simple to use, we assume that the primary target would be educated, technology-affine people acting as opinion leaders. Contrary to other solutions, we do not require TPM modifications like additional interfaces. We use visual communication to establish an authentic channel to the system the user is interested in. Opposed to other visual approaches, no barcode stickers on the devices are required which eliminates the risk of manipulated stickers. With our prototype implementation, we demonstrated the

[2]TrouSerS:http://trousers.sf.net/ (visited Nov. 2009)

[3]TPM Emulator / Infineon SLB9635TT / Atmel AT97SC3203

feasibility of our approach and evaluated the expected performance impact on a smart camera.

There is a number of open issues to be addressed in future work. We currently assume that the user's handheld is a priori trusted and honestly displays the attestation result to the user. Depending on the requirements, the trustworthiness of the handheld however needs be attested separately. Another issue are potential denial of service (DoS) attacks by repeated attestation requests. We currently protect our prototype from overload by only accepting request at a predefined time interval. A more robust DoS attack protection could be based on attestation tickets issued by camera network operators. Furthermore, our current implementation is limited to attaching simple properties to processing blocks. In a more holistic approach, also the configuration parameters of the processing blocks need to be measured, evaluated by the TTP and included in the reported set of properties.

In conclusion, we believe that Trusted Computing can be a valuable component for building secure and trustworthy camera networks. With our current work we have specifically addressed the interests of users. We allow users to query the state of a camera and derive a trust decisions whether their privacy is protected or not. While this is an important step, it still has several shortcomings. First, users need some basic knowledge about the involved mechanisms to understand the outcome of the attestation. Second, people have no influence on what cameras are doing. Ideally users should not only be able to check that state of cameras but also, to a certain extend, be able to influence the behaviour of cameras. One of the key questions here is to find a tradeoff such that the user's interests are satisfied and the camera system still remains usable for the intended purpose.

REFERENCES

[1] A. Cavallaro, "Privacy in Video Surveillance," *IEEE Signal Processing Magazine*, vol. 24, no. 2, pp. 168–166, March 2007.

[2] S. Soro and W. Heinzelman, "A Survey of Visual Sensor Networks," *Advances in Multimedia*, vol. 2009, pp. 1–21, May 2009.

[3] S. Moncrieff, S. Venkatesh, and G. A. W. West, "Dynamic Privacy in Public Surveillance," *IEEE Computer*, vol. 42, no. 9, pp. 22–28, Sep. 2009.

[4] N. Baaziz, N. Lolo, O. Padilla, and F. Petngang, "Security and Privacy Protection for Automated Video Surveillance," in *Proceedings of the IEEE Int. Symposium on Signal Processing and Information Technology*, 2007, pp. 17–22.

[5] A. Chattopadhyay and T. Boult, "PrivacyCam: A Privacy Preserving Camera Using uCLinux on the Blackfin DSP," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007, pp. 1–8.

[6] F. Dufaux and T. Ebrahimi, "Scrambling for Video Surveillance with Privacy," in *Proceedings of the Computer Vision and Pattern Recognition Workshop*, 2006, pp. 160–166.

[7] T. Winkler and B. Rinner, "Applications of Trusted Computing in Pervasive Smart Camera Networks," in *Proceedings of the Workshop on Embedded System Security (WESS)*, 2009.

[8] D. N. Serpanos and A. Papalambrou, "Security and Privacy in Distributed Smart Cameras," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1678–1687, October 2008.

[9] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Y.-L. Tian, A. Ekin, J. Connell, C. F. Shu, and M. Lu, "Enabling Video Privacy through Computer Vision," *IEEE Security & Privacy Magazine*, vol. 3, no. 3, pp. 50–57, May/June 2005.

[10] S. Tansuriyavong and S. Hanaki, "Privacy Protection by concealing Persons in circumstantial Video Image," in *Proceedings of the Workshop on Perceptive User Interfaces*, 2001, pp. 1–4.

[11] B. Parno, "Bootstrapping Trust in a "Trusted" Platform," in *Proceedings of the Usenix Workshop on Hot Topics in Security*, 2008.

[12] R. Toegl, "Tagging the Turtle: Local Attestation for Kiosk Computing," in *Advances in Information Security and Assurance*, 2009, pp. 60–69.

[13] J. M. McCune, A. Perrig, and M. K. Reiter, "Seeing-Is-Believing: Using Camera Phones for Human-Verifiable Authentication," *Int. Journal of Security and Networks (IJSN)*, vol. 4, no. 1/2, pp. 43–56, 2009.

[14] S. Garriss, R. Cáceres, S. Berger, R. Sailer, L. van Doorn, and X. Zhang, "Trustworthy and Personalized Computing on Public Kiosks," in *Proceedings of the Int. Conf. on Mobile Systems, Applications, and Services*, 2008, pp. 199–210.

[15] E. Bangerter, M. Djackov, and A.-R. Sadeghi, "A Demonstrative Ad Hoc Attestation System," in *Proceedings of the Int. Conf. on Information Security (ISC)*, 2008, pp. 17–30.

[16] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun, "Loud and Clear: Human-Verifiable Authentication Based on Audio," in *Proceedings of the IEEE Int. Conf. on Distributed Computing Systems (ICDCS)*, 2006.

[17] *TCG Software Stack Specification (TSS) Version 1.2, Level 1, ErrataA*, Trusted Computing Group Std., March 2007.

[18] *TPM Main Specification Version 1.2, Level 2, Revision 103*, Trusted Computing Group Std., July 2007.

[19] M. Pirker, R. Toegl, D. Hein, and P. Danner, "A PrivacyCA for Anonymity and Trust," in *Proceedings of the Int. Conf. on Trusted Computing (TRUST)*, 2009, pp. 101–119.

[20] M. Strasser and H. Stamer, "A Software-Based Trusted Platform Module Emulator," in *Proceedings of the Int. Conf. on Trusted Computing and Trust in Information Technologies (TRUST)*, 2008, pp. 33–47.

[21] A. Mohan, G. Woo, S. Hiura, Q. Smithwick, and R. Raskar, "Bokode: Imperceptible Visual Tags for Camera Based Interaction from a Distance," in *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, 2009, pp. 1–8, article No. 98.