Inverting Sensor Networks and Actuating the Environment for Spatio-Temporal Access Control

Shu Chen, Yu Zhang, Wade Trappe
Wireless Information Network Laboratory (WINLAB)
Rutgers, The State University of New Jersey
671 Rt. 1 South
North Brunswick, NJ 08902
shuchen@cs.rutgers.edu; yu, trappe@winlab.rutgers.edu

ABSTRACT

Wireless sensor networks are typically deployed to measure the information field, rather than create an information field. However, by utilizing the radio on sensor nodes, it is possible to invert the role of sensor networks, and allow sensor nodes to actuate the environment. Such actuation can facilitate new forms of access control that are based on whether a user is located at the right place at the right time. In this paper, we explore the challenges of supporting spatio-temporal access control, where access to an object or service is based on the user's spatio-temporal context. Specifically, we focus on supporting spatio-temporal access control through the specification of access control policies, and show how complex spatio-temporal policies can be specified using automata. We outline a challenge-response mechanism for verifying user location in a centralized spatio-temporal access control mechanism. We utilize sensor networks in an inverted fashion to support spatio-temporal access control. Sensor nodes announce keys according to a time-varying schedule, and users may access restricted files/resources only if they are in the neighborhood of the correct sensor node and witness the appropriate cryptographic key.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: [distributed networks, network communications]

General Terms

Security

Keywords

Access Control, Localization

1. INTRODUCTION

Historically, wireless networks have freed users from the confines of static, wired networks, and traditionally access control mechanisms are not based upon the geographic properties associated with the wireless user. The fact that wireless networks are becoming increasingly ubiquitous, however, suggests that it is not necessary to restrict access to services based solely on conventional identity-based authenticators. Rather, the wireless infrastructure can facilitate location-aware computing paradigms, where services are only accessible if the user is in the right place at the right time. For example, location-aware security services, such as ensuring that a file can only be accessed within a specific secure room, or that a laptop no longer functions when it is taken outside of a building, are not only desirable but will soon become feasible.

Wireless sensor networks are traditionally used to monitor phenomena and record data corresponding to the underlying physical information field. The small form factor and low-power design associated with many sensor architectures, suggests that this inherently pull-oriented use of sensor networks can be turned upside down to push information into the physical environment over extended periods of time. By using sensor nodes themselves to transmit signals into the wireless medium, it is possible actuate the physical environment to convey information useful for new types of mobile services. This notion of inverting the network can be used to facilitate new forms of access control that can now be based upon whether an entity is at the right place at the right time in order to witness the information needed to access content.

In this paper, we propose the use of "inverted" sensor networks to provide access control based upon a user's spatio-temporal information. Access control systems involve two main components: the security policy, which is a statement of who is allowed to access which service; and security mechanisms, which are the methods by which the security policies are enforced. Therefore, in this paper, we will explore both of these aspects in the context of using sensor networks for spatio-temporal access control (STAC). This paper explores both of these components by providing a formalism for spatio-temporal access control (STAC) models, as well as key distribution issues associated with spatio-temporal access control utilizing sensor networks.

We begin the paper in Section 2 by providing a brief overview of inverting sensor networks and their use in achieving spatio-temporal access control. Following this high-level description, we will explore both the specification and enforcement of STAC policies in more detail. In Section 3, we describe the components involved involved in STAC and provide several different means to describe STAC policies. In particular, we show that it is possible to achieve more flexible representations of STAC policies through the use of finite automata. We briefly examine how STAC can be achieved using a centralized entity, and propose a location-oriented challenge response protocol to support STAC. However, we believe that centralized schemes can expose a user to a severe privacy risk, and therefore, in order to achieve better user privacy, in Section 5, we turn our focus to the problem of using the sensor network to enforce STAC policies. By having sensor networks appropriately announce keys at the correct time, it becomes possible to confine the access to content objects to specific spatio-temporal regions. We then move onto placing our work in the context of the broader access control literature in Section 7. Finally, we conclude the paper and discuss further directions we are currently investigating in Section 8.

2. OVERVIEW OF INVERTED SENSOR NETWORKS

Traditionally, wireless sensor networks have been used for a variety of monitoring applications, ranging from military sensing to industry automation and traffic control. A common feature to all of these different sensor applications is that they *pull* data from the environment, i.e. they collect data and route this data (or process locally) for external applications to utilize. This pull-oriented formulation is the original, intuitive usage of sensors nodes. However, since sensor nodes consist of radios that they use for communication, it is also possible to turn the the role of the sensor node upside down and make the sensor *push* information into the environment.

This notion of an inverted sensor network is contrary to the traditional application of sensor networks where the sensors themselves merely gather data that another device can use to actuate and make decisions with. In the inverted model, however, the environment is changed using the radio, and a new information field is created in RF space that may be monitored using other devices. Due to the low-power nature of sensor node transmissions, the information injected into the environment by a sensor node is inherently localized.

The localized effect that a sensor node can have on the RF environment makes it possible to support access control based on the spatio-temporal location of a node that monitors the environment. Spatio-temporal access control allows for objects to be accessed only if the accessing entity is in the right place at the right time. STAC may thus be supported by having the environment locally convey the cryptographic information (keys) needed by the entity in order to access enciphered objects. To achieve STAC, then, a sufficiently dense network of low-power radios, as exists in traditional sensor nodes, is needed, and each sensor node can transmit a time-varying schedule of assigned cryptographic information in support of STAC.

While conventional access control is based on the user's identity, there are many scenarios where identity-based authentication is not only inconvenient but also unnecessary, and instead user spatio-temporal contexts are more desirable for basing access control upon:

1. A company may restrict its commercially confidential documents so that they can only be accessed while inside a building during normal business hours.
2. Network connectivity may be provided only to users who are located in a specific room (e.g. a conference room) during a specified meeting time.
3. Devices, such as corporate laptops, can be made to cease functioning if it leaves the building.
4. During sporting events, a sports service may transmit value-add information, such as live scores and player information, during the game, but only want those within the stadium to be able to access this information.
5. Movies or entertainment may be made to be accessible only to vehicles that are located on a specific road.

The above examples illustrate cases where both objects and services may be restricted based on location. Although the implementation of STAC to services, such as network connectivity, might not necessitate use of encipherment, we shall take the viewpoint in this paper that all objects/services that we wish to control access to can be suitably protected through encryption and appropriate key management. The extension of the ideas provided in this paper to more general cases, such as controlling access to network connectivity, can be done through simple resource allocation mechanisms, and we thus consider general cases a straight-forward application of the techniques detailed in this paper.

Throughout the rest of this paper, we shall refer to the STAC communication model depicted in Figure 1. In this model, we have an object that is protected through encryption with a set of keys. These keys may, for example, encrypt different portions of the object. Assisting in the STAC process is a broad array of sensor nodes, spread out over the region of interest, that each hold a set of decryption keys. These sensors can be configured to emit specific keys at specific times with specific power levels. As a result, only users that are near the right sensor at the right time can witness keys and access objects. The schedule of key transmissions, and their corresponding power levels, can either be pre-loaded, or controlled by an external entity connected to the sensor network, as depicted in the figure. Additionally, in the STAC model depicted in the figure, we present two different means by which the user can obtain a protected object: first, it can be broadcasted via a central entity (much like a television broadcast); or it can downloaded, locally stored on the user's device, and then accessed when the device witnesses the appropriate keys.

3. SPATIO-TEMPORAL ACCESS CONTROL MODEL

In this section, we capture the essential features of a STAC system and present a basic STAC model. The pieces described here serve as an underlying framework for our later discussion where we use inverted sensor networks to achieve STAC.

3.1 STAC Components

Analogous to the primary features of the RBAC model, as described in [1], STAC involves five basic elements: *users* (USERS), *objects*(OBS), *operations* (OPS), *spatio-temporal regions* (STRGNS), and *permissions* (PRMS). Figure 2 illustrates the core components of the STAC model. A set of STAC *access policies* describes rules that specify what permissions/privileges may be granted/rejected based upon a

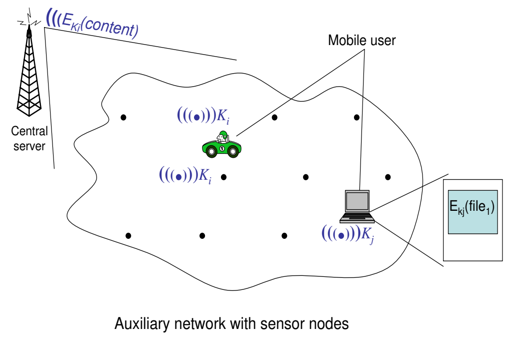

Figure 1: Our basic architecture for spatio-temporal access control consists of a central entity that supplies encrypted content, an auxiliary network of sensor nodes that emit keys, and mobile users that desire to access content based on their spatio-temporal context.

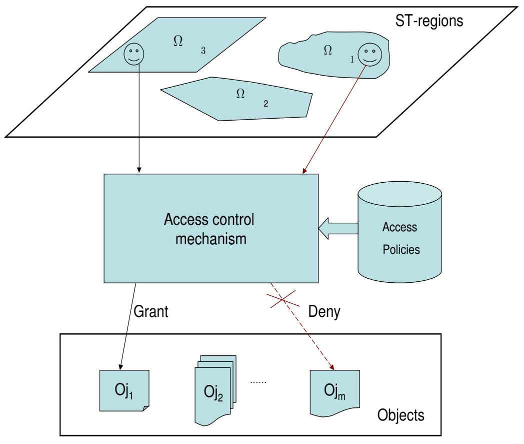

Figure 2: An conceptual picture of STAC model

user's access attempt. The central idea of STAC is to control the *permission* or rejection of a *user*'s *operation* on an *object* based on the *spatio-temporal region* the user is in, according to these *access policies*. *Access control mechanisms* are the collection of techniques (e.g. encipherment) used to enforce this process. We now specify each of these components in more detail.

User. A *user* is generically defined as any entity, such as a process or person, that seeks access to objects.

Operation. An *operation* is a function that a user can execute on an object. The types of operations depend on the types of applications and systems. For example, when considering access to a file, the operations could either be simply access/no-access, or they can be multilevel such as read, write or execute. For the simplicity of discussion, we shall restrict our attention to binary operations. We note, however, that multilevel cases can be converted to the binary cases by defining a set of new objects, one for each operation, that substitute for the original one. For example, given an object *f1* with 3 operations read, write or execute, we define 3 new objects *f1read, f1write, f1exec* that substitute for the original object *f1*, and each of these new objects becomes a binary case. Throughout this paper, we shall represent access privileges for binary operations by 1 corresponding to access approval, while 0 corresponds to access rejection.

Permission. The set of *permission* is defined as PRMS $\subseteq 2^{(OPS \times OBS)}$, where OPS is the set of operations and OBS is the set of objects. A permission is an approval of an operation on an object. Since the operations discussed here are binary, each permission *prms* then has the form $p \in \{0,1\} \times OBS$.

Object. An *object* can be an information container, such as a file or an exhaustible system resource (e.g. a wireless service or CPU cycles). As stated in the last section, we assume the objects can be protected through encryption and appropriate key management. When an object is involved in the STAC system, it is endowed with temporal character, and can be generally categorized into static and streaming cases. A streaming object continually evolves with time, such as a movie being broadcast to an entire network of users, or live scores from a sporting event being transmitted to the audiences in a sports arena. On the other hand, a static object does not change with time (except for the possible exception of version revision, as often occurs for software objects). Since streaming objects are time-varying, their corresponding access control policies will inherently become more complicated to express.

As an example, consider an object O_j that is a streaming object. We may partition this object into subobjects according to time intervals, e.g. $O_j = \{O_j[t_0, t_1), O_j[t_1, t_2), O_j[t_2, t_3)\}$. Here O_j has been broken down into three pieces according to the time intervals $[t_0, t_1)$, $[t_1, t_2)$, and $[t_2, t_3)$ respectively. These subobjects may be further decomposed, and such a decomposition naturally raises the issue of the maximum amount that an object can be decomposed. We refer to the smallest constituent piece of a larger object as the *object atom*. The size of an object atom is determined by the *temporal resolution* of a STAC system.

Spatio-temporal region. A useful concept for specifying STAC for streaming objects is the notion of a *spatio-temporal region*. A spatio-temporal region, denoted by Ω is defined as a set of 3-tuple, $\Omega = \{(x, y, t)$: valid areas in space and time$\}$, where (x, y) represents a spatial location and t represents an arbitrary time instance, and hence each (x, y, t) is a point in 3-dimensional spatio-temporal space. The spatio-temporal regions, which we shall refer to as ST-regions for brevity, are set up by the system according to the access policies, such as which object can be accessed where at what time period. Thus, it is often useful to visualize a ST-region as a continuous region in 3-dimensional space, instead of a set of discrete points. Figure 3 shows two example ST-regions Ω_1 and Ω_2. Ω_1 indicates a spatial region that is constantly specified from time 0 to time t; whereas Ω_2 indicates a spatial region that varies with time and, hence, if associated with an object's spatio-temporal access policy, would require that a user must move in a specific manner in order to maintain access privileges to an object. For an object *ob* and an operation *op*, a ST-region is called the *secure*

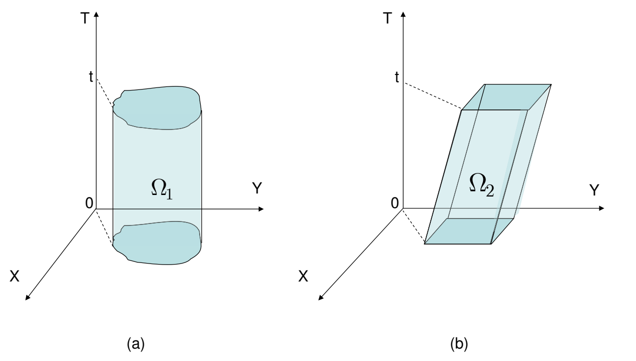

Figure 3: Examples of two spatio-temporal regions Ω_1 and Ω_2.

ST-region of (ob, op) if the operation *op* is allowed to be performed on the object *ob* at this ST-region. In the case that the operation is binary, as we focus on in this paper, we simply refer to such a region as the *secure ST-region of ob.*

3.2 Access policies and their representations

3.2.1 Basic access policies and their representations by an access control matrix

Access policies outline the rules and regulations for appropriately accessing the objects. In STAC, a basic access policy is a 3-tuple $A = \{(\Omega; op; O_j;)\}$, where $O_j \in OBS$, $op \in OPS$, $\Omega \in STRGNS$, which is interpreted to mean that within the ST-region Ω, the operation *op* on object O_j is approved. Access control matrices can be used to represent such access policies. In the access control matrix, the columns correspond to objects, the rows to ST-regions, and the cell where the column and row intersect specifies the access privilege. Table 1 shows an example of an access control matrix for STAC, where the operations on objects are binary.

In this table, for example, object $S1$ can be accessed in the spatio-temporal location Ω_1, but not in Ω_2 (which has been further decomposed into smaller spatio-temporal regions). As a more involved example, we can decompose object Mv (which might correspond to a movie) down into sub objects (e.g. first 10 minutes, second 10 minutes, etc.). By similarly decomposing region Ω_2 into smaller regions Ω_{21}, Ω_{22}, Ω_{23} and Ω_{24}, we may now describe a more refined STAC policy, where Mv_1 is accessible at location Ω_{21} but not Ω_{22}, and thus the user must move its physical location with an appropriate time period to Ω_{22} in order to access Mv_2, and hence resume access to Mv.

3.2.2 Complex access policies and their representations by FA

The example of object Mv just described gives insight into the power of STAC. In particular, a STAC system can perform complex access control by decomposing objects into *object atoms* and suitably associating smaller *region atoms* with these objects. By doing so, it is possible to grant or deny a user access to an object not only based on his current location, but also based on his previous spatio-temporal behavior. For example, we might require (for additional security), that a user have the ability to access object ob_1 at location l_1 and time t_1, and then be at location l_2 at time t_2 in order to access object ob_2. That is, without having been to (l_1, t_1) and having accessed ob_1, the user would not have the requisite access control information needed in order to access ob_2 at location (l_2, t_2).

Such a form of access control is stateful, and is unwieldy for representing with access control matrices. Notice in Table 1, for example, that although object Mv_1 can be accessed at Ω_{21}, Mv_2 can be accessed at Ω_{22}, and Mv_3 can be accessed at Ω_{23}, there is no information contained in the matrix that specifies that the user had to be at Ω_{21} before proceeding to Ω_{22} in order to access Mv_2.

In order to represent these more advanced forms of STAC policies, we must employ a syntactical framework that facilitates the representation of state. We now show how automata theory can be employed to describe such complex polices. First, though, we provide a brief review of automata theory to facilitate our later discussion. A *finite automaton* is denoted by a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of *states*, Σ is a finite *input* alphabet, q_0 in Q is the *initial* state, $F \subseteq Q$ is the set of *final* states, and δ is the *transition function* mapping $Q \times \Sigma \to Q$ [2]. A string x is said to be *accepted* by a finite automaton $M = (Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = p$ for some $p \in F$. The *language accepted by M*, designated $L(M)$, is the set $\{x|\delta(q_0, x)\}$.

We may represent an access policy using an automaton $M = (Q, \Sigma, \delta, q_0, F)$ and capture the user's history for consideration in the access policy. To do so, we let the input alphabet $\Sigma = STRGNS \cup OBS$. A string x that is accepted by M is a sequence mixed by ST-regions that a user is required to locate and the objects that the user is required to access. By properly designing the δ and the set of states Q, the desired access policy can be expressed.

In order to illustrate how, we now provide an example. Suppose that we have a movie that is divided into 3 parts—Mv_1, Mv_2 and Mv_3. Further, suppose that the access policy we want to describe is that in order for a user to be able to view a later part of the movie, he must have finished viewing the part(s) that came before that part. Further, suppose that we have the additional spatio-temporal requirement that Mv_1 can only be accessed at location l_1 at time t_1, Mv_2 can only be accessed at location l_2 at time t_2, and Mv_3 can only be accessed at location l_3 at time t_3.

A finite automaton M_1 for this policy is defined as follows. $M_1 = (Q, \Sigma, \delta, q_0, F)$, where $Q = \{q_0, q_1, q_2, q_3, p_1, p_2, p_3\}$; $\Sigma = \{\Omega_1, \Omega_2, \Omega_3, Mv_1, Mv_2, Mv_3\}$; $\Omega_1 = (l_1, t_1)$, $\Omega_2 = (l_2, t_2)$, $\Omega_3 = (l_3, t_3)$; $F = \{p_1, p_2, p_3\}$. Here, in our specification, there are two classes of states q_j and p_j. The states q_j correspond to a description of the user being at the spatio-temporal contextual state needed to access the movie object Mv_j, where as state p_j corresponds to a description that the user has been in state q_j, and then performed the required accessing of the movie object Mv_j (and consequently, can move to the next ST-region Ω_{j+1}). If we examine the transition diagram depicted in Figure 4, we can describe the transition mapping as follows:

$$\delta(q_0, \Omega_1) = q_1$$

$$\delta(q_1, Mv_1) = p_1$$

Table 1: Access Control Matrix of STAC

		S1	**S2**	**F1**	**Mv**			**...**	Oj_m	
		-	-	-	Mv_1	Mv_2	Mv_3	...	Oj_{m_1}	Oj_{m_2}
Ω_1	-	1	1	0	0	0	0	...	0	0
Ω_2	Ω_{21}	0	1	1	1	0	0	...	0	0
	Ω_{22}	0	1	1	0	1	0	...	0	0
	Ω_{23}	0	1	1	0	0	1	...	0	0
	Ω_{24}	0	1	1	0	0	0	...	0	0
...	...	...	...	...	...	...	...	...	...	...
Ω_n	Ω_{n1}	1	0	0	1	1	1	...	1	0
	Ω_{n2}	1	0	0	1	1	1	...	0	1

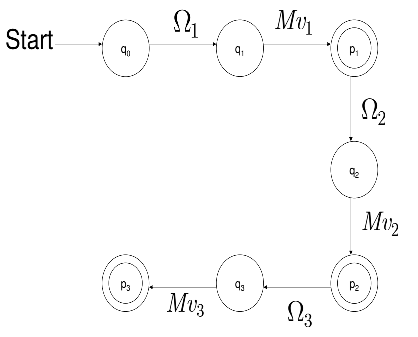

Figure 4: The transition diagram for M_1

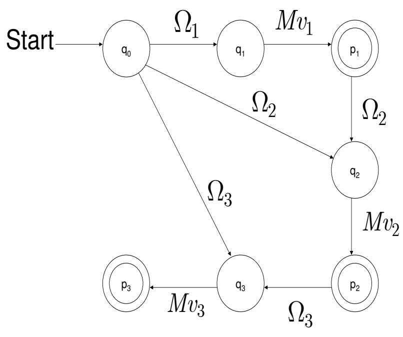

Figure 5: The transition diagram for $M_1^{'}$

$$\delta(p_1, \Omega_2) = q_2$$

$$\delta(q_2, Mv_2) = p_2$$

$$\delta(q_2, \Omega_3) = q_3$$

$$\delta(q_3, Mv_3) = p_3.$$

We now walk through this transition mapping. The user starts in a null state q_0, and if it has moved to location Ω_1 at time t_1, it is described as being in state q_1, and hence has the ability to access Mv_1. Once the user has watched the movie portion Mv_1, its state transitions to p_1, and the user may now move to location Ω_2 (by time t_2). The process continues, as the user moves to a new location, its state changes so it can access the content, then having accessed the content the user can now move to the next location. Formally, we may state the accepted language of this automaton M_1 as the collection of valid strings: $L(M_1) = \{\Omega_1 Mv_1, \Omega_1 Mv_1 \Omega_2 Mv_2, \Omega_1 Mv_1 \Omega_2 Mv_2 \Omega_3 Mv_3\}$.

Finite automata are fairly flexible and able to represent stateful access control policies rather easily. As another example, the finite automaton M_1 can be modified easily to support similar but slightly different policies. For example:

- If we add two transition functions to M_1. $\delta(q_0, \Omega_2) = q_2$, $\delta(q_0, \Omega_3) = q_3$, as shown in the transition diagram for the finite automaton M_1' in Figure 5, then the STAC policy takes on a different interpretation. Now, the STAC policy does not explicitly require that the user must have accessed the prior content Mv_{j-1} before accessing Mv_j. Instead, all that is required is that the user is at Ω_j in order to access Mv_j. This scheme corresponds to the movie access policy specified in the access control matrix in Table 1.
- We note that it is also possible to specify policies that have no involvement with the object, but instead only require the user to go through a particular spatio-temporal path. Such policies, as depicted in the transition diagram in Figure 6 for an automaton M_1'', might not be directly interesting from an access control point of view, but they can be useful as building blocks for more complicated STAC policies. For example, one can easily envision requiring that a user go to a succession of different locations, prior to being able to access content.

4. CENTRALIZED METHODS FOR STAC

Access control mechanisms are the means to enforce access policies. There are usually two steps necessary for supporting spatio-temporal access control: encipher objects to prevent unauthorized access to objects; and ensure that only users at an appropriate spatio-temporal location can acquire the keys needed to decipher objects.

Before delving into using inverted sensor networks to achieve STAC, we note that it is possible to provide access control to objects by holding the objects and then require that an entity prove that it is at a specific location before distributing the object to that entity. Or, as another centralized approach, an entity can have an enciphered object and, upon

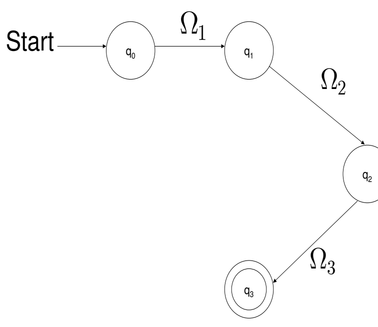

Figure 6: The transition diagram for $M_1^{''}$

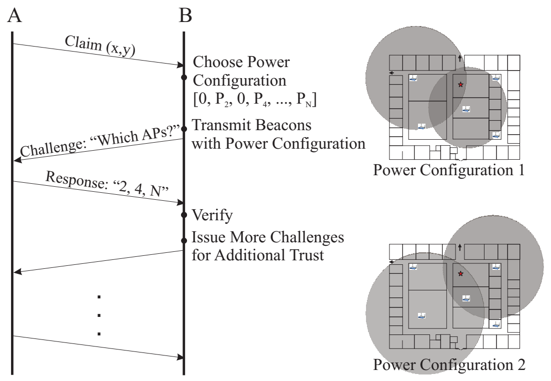

Figure 7: Location verification based upon inclusion principle and the notion of power modulated challenge-response.

proving that it is in a specific location, the central server can deliver the set of keys needed to decipher and access the object.

In order to facilitate such centralized approaches for STAC, it is necessary to have trustworthy location information. Conventional localization schemes [3–6] can provide location information, but recent efforts by many researchers have revealed that these localization algorithms can be attacked/subverted by non-cryptographic mechanisms. In response to this weakness, there has been a concerted effort to develop secure positioning techniques, where the localization algorithm is robust to measurement attacks or impersonation [7–9].

Although these techniques can reliably localize an entity, we feel that such methods don't reflect the natural operation of an authentication protocol used in access control. Instead, we now present a different form of the locationing problem in which we have a mobile node *claiming* that it is at a particular location, and we desire the centralized entity to *authenticate* that claim [10]. Our location verification scheme, depicted in Fig. 7, involves a *challenge-response* protocol, and uses variable power configurations for the underlying wireless network to corroborate the claimed location.

Suppose we have an infrastructure of anchor points AP_j of known locations (x_j, y_j), where $j = 1, 2, ..., K$, capable of emitting localization beacons. Suppose that a mobile device contacts the infrastructure, claiming that it is at a location (x, y). It is the task of the infrastructure to validate this claim. To do so, it will issue a *challenge* to the mobile by creating a random test power configuration intended to verify the claimed location. This power configuration corresponds to the powers used by the different access points when transmitting locationing beacons. The power configuration will involve a power of 0 for some access points, meaning that these APs do not transmit, while a power of P_j is chosen for other APs. The powers P_j are chosen to define a radio region about AP_j so that the node *should* be able to witness the beacon from its claimed position (x, y). The determination of a radio region Ω_j is done using a propagation model.

The infrastructure now sends the challenge "Which APs do you hear?" to the mobile. The power levels of the APs are temporarily adjusted and location beacons are issued. The mobile then responds with a list of the APs it was able to witness, and the infrastructure checks this response. If a device incorrectly reports that it heard an AP that was not present, then this is clear evidence that the device's truthfulness, and hence its position, is false. However, if a device reports some APs correctly, but fails to report an AP that it should have heard, then we do not conclude the device's location is false. Rather, it may be that the beacon was simply missed due to poor propagation. We can assert the likelihood that a device misses a beacon using the underlying propagation model, and incorporate this confidence measure into verifying the device's location. In order to enhance the confidence levels of the claimed location, the challenge-response process may be repeated several times with different configurations.

We now provide a basic analysis of the performance of the power-modulated challenge response protocol under two scenarios: the adversary is not able to witness any AP during the challenge, and a legitimate device is truly where it claims to be located. We will suppose that there are K APs in the WLAN, and that each round of the protocol we randomly choose k APs and set their power levels to provide a 95% coverage guarantee of the claimed (x, y) location. If the adversary knows that k APs were used but cannot witness any APs, then there is a probability of $\binom{K}{k}^{-1}$ of correctly guessing the APs. If the procedure runs N protocol rounds, then the probability of the adversary incorrectly being authenticated is $p_a = \binom{K}{k}^{-N}$. On the other hand, if the device is legitimate and at the correct location, the probability that it successfully witnesses all k APs is $(0.95)^k$. Thus, the probability of a legitimate device failing to authenticate in N protocol rounds is $p_d = 1 - (0.95)^{kN}$. Ideally, both p_a and p_d should be small. This implies that there is a fundamental tradeoff between K, k, and N in such a scheme. We note, however, that the above analysis only captures two extreme cases, and does reveal the true complexities associated with the authentication problem. In particular, the analysis does not reflect the possibility of an adversary capable of witnessing some APs and hence who possesses partial knowledge of the challenge pattern. The primary focus of this paper, though, is on our proposed decentralized mechanisms for STAC, and we will thus provide further analysis of centralized challenge-response location verification techniques in an expanded follow-up work.

One drawback of centralized STAC techniques is that the interaction between a user and the infrastructure inherently introduces an issue related to the user's privacy. In particular, a user's location information becomes exposed in centralized STAC mechanisms, and consequently users may be tracked by the infrastructure [11]. Although there have been some efforts recently that have examined location privacy, e.g. [12], these efforts are primarily focused on services that provide location traces to other services, and not on privacy issues associated with the measurement of a transmitter's location.

We will revisit the privacy issue in a later discussion. However, in the next session, we present our inverted sensor network construction, which achieves user location privacy by not requiring interaction between a user and the network infrastructure.

5. DECENTRALIZED STAC THROUGH INVERTED SENSOR NETWORKS

In this section, we describe the use of inverted sensor networks to support spatio-temporal access control. In the basic inverted sensor network scheme, we assume that sensor networks are deployed in a regular (lattice) pattern, and that the sensors employ constant power levels when transmitting keys. This leads to a coverage issue, and further raises the issue of how precisely we may cover a specified spatio-temporal region Ω using the basic configuration. Since the general sensor network will not be deployed in a regular pattern, and since sensor nodes can employ variable power levels across the network, we next examine the issue of adjusting the coverage region to support a spatial region. Finally, we examine issues related to key deployment/management and the frequency of key announcement in order to support a desired time resolution for the region Ω.

5.1 Inverted sensor network infrastructure

There are three basic components involved in the inverted sensor network approach to STAC, as shown in Figure 1. The first component is a centralized content distributor that does not need to have any interaction with the user. Instead, the content distributor might broadcast or supply objects for download that have been enciphered with a set of keys known by the central server.

The second component is the auxiliary network (the inverted sensor network), which consists of sensor motes that have been deployed to cover a region of interest. These sensor motes will transmit a schedule of encryption keys that vary with time. Here, we assume that time is broken down into intervals, and that during any given interval, the corresponding key will be repeatedly transmitted at regular intervals. Any entity within the radio range of the sensor mote can, should it wish, acquire the key that is announced by that sensor at that time. We note that the keys that are transmitted by the sensor motes must be initially distributed to these nodes for use in supporting STAC.

Finally, the third component is the mobile user itself, which must move around the region of interest in an appropriate manner in order to acquire the keys transmitted by the sensor nodes.

In order to explore the properties of the inverted sensor network, we assume that sensor nodes are deployed in a regular, hexagonal fashion across the region of interest. It is well known that a regular hexagonal lattice is more efficient than either a rectangular or quincunx sampling in two-dimensions. Hence, we assume the coverage area is partitioned into hexagon cells of the same size, and that a single sensor node is placed at the center of each hexagon. Prior to initiating the STAC enforcement, we assume that each sensor has been assigned a schedule of keys (for simplicity, we shall say that the keys have been distributed by a centralized key distribution center). During STAC operation, each sensor emits a key according to its schedule, which can be observed by any other entity within radio range of the sensor.

If we let a be the length of an edge of one of the regular hexagons, and assume isotropic radio propagation then we may assume that each radio coverage is a circle with radius r. There are two natural choices for how the hexagonal lattice is deployed: either we deploy the sensor nodes so that the radio ranges do not share any overlap (and hence the circular regions would touch each other only at tangent points), or we can assume that we have deployed the sensors so that there is some overlap between the radio regions. Since the first option implies gaps in the radio coverage of the sensor grid, we assume that the deployment is of the second type. In this case, we have a deployment such as the one depicted in Figure 8, and hence the radius of radio coverage is $r = a$.

Given a ST-region of an object that needs to be protected, we assume that the content distributor knows the key schedule of all of the sensor nodes (perhaps it is also the key distributor and generated the key schedule in the first place), and that the object has been encrypted with the symmetric encryption key corresponding to the key that is being transmitted by a specific sensor node at a specific time. Exploring this idea further, reveals a some system requirements. If we need a STAC region that is larger than a single radio region of a sensor node, then we need all sensor nodes within the ST-region to transmit the same key. This requires that either the key distribution center has a priori knowledge of the ST-regions (perhaps corresponding to one or more objects that need to be protected) that need to be enforced for spatio-temporal access control, or that the key distribution center has a means to adapt the key transmission schedule of the sensor network. As an example, Figure 8 illustrates the key distribution scheme for two secure ST-regions Ω_1 for object O_1 and Ω_2 for object O_2. Here, we suppose k_1 and k_2 are the decryption keys for O_1 and O_2 respectively at a particular time. Therefore, it is necessary that k_1 has been assigned to all sensors whose radio discs are inside the rectangle Ω_1, and k_2 to the sensors whose radio discs are inside the rectangle Ω_2.

5.2 Improving the coverage

As seen in Figure 8, the regions of Ω_1 and Ω_2 are not fully covered by the keys sent by sensors. In particular, the concept of an *approximating ST-region* naturally arises.

Definition 1: An *approximating ST-region* $\overline{\Omega}$ of a ST-region Ω given an STAC mechanism Σ is the spatio-temporal region that the object is actually accessible under Σ.

In practice, the approximating ST-region is not likely to be the same as the original, desired ST-region, and in fact we are only able to protect an approximation of the original ST-region. If we restrict our attention to just the spatial portion of a ST-region (which we shall denote by R), then we want the approximating ST-region to cover as large of a portion of the desired ST-region as possible. Before we proceed onto exploring how to optimize the approximating ST-region, we note that our discussion has centered around approximating a ST-region from the inside, and that it is possible to consider approximating regions that are *larger*

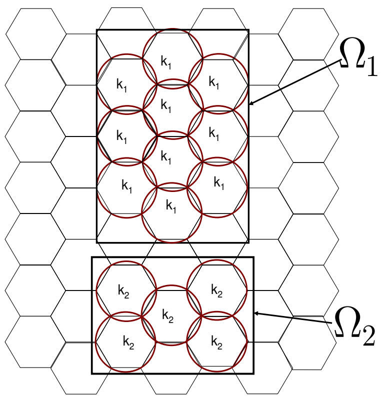

Figure 8: An example of how the keys may be assigned in order to cover two ST regions Ω_1 and Ω_2 for an inverted sensor network support spatio-temporal access control.

than the ST-region that they are meant to represent. In such a case, we include sensors whose radio regions share any overlap with the desired ST-region. For this case, we would aim to reduce the amount of extra area covered by the ST-region.

For this work, though, we focus on approximating a ST-region from within, and hence we would like to minimize the amount of blank area between a desired ST-region and the approximating ST-region. Although we have deployed our sensors in a regular fashion, we may adjust the transmission powers of the sensor nodes in such a way so as to improve the amount of area covered by the radio regions.

To formalize this, let us define the region of interest to be G, and define $S = \{s_j\} \in G$ to be a set of sensor nodes. Here, we use the notation s_j to refer to the spatial position of node j. In the discussion that follows, we do not require that the sensor positions fall on a lattice, but instead the positions can be more general. For a given region $R \subseteq G$ we may specify what it means to cover (or fill) R from within.

Definition 2: A *cover of R from inside*, denoted C, is a set of circles C_j centered at s_j such that the union of the circles is fully within the inside R, that is $C = \bigcup(C_j) \subseteq R$. A cover from inside C is a function of a subset of the sensor nodes that are selected, and the corresponding transmission powers assigned to each of these sensor nodes. Hence, if we denote $\underline{P} = \{p_1, p_2, ...\}$ to be the power allocation vector for nodes $\{s_j\}$, then we may represent the cover as $C(\underline{P})$.

A cover from the inside will typically not completely cover the region R in the formal topological sense, and thus we are interested in measuring how accurately a cover from the inside approximates R. To capture this notion, we introduce the blank region and its corresponding area.

Definition 3: For a cover from inside C of a region R, a *Blank Region* is the set $B(C,R) = R \setminus \bigcup(C_j^{'})$. A measure of the magnitude of the blank region is the *Blank Area*, $BA(C,R) = Area(R \setminus \bigcup(C_j^{'}))$.

In an access control system, it is desirable to minimize the blank area. We note that it is easily possible to minimize the blank area by covering areas outside of R. This, however, implies that users who are not in the restricted area can

```
Data: Spatial Region $R$, the set of sensor node locations $s_1, s_2, ..., s_n$ and the maximal radius constraint $m$
Result: An cover from inside $C$, such that $BA(C,R)$ is minimal.
for Every $s_i$ inside of $R$ do
    Draw an inscribed circle centered at $s_i$, $r_i$ denotes the radius of the inscribed circle.;
    $p_i$ = min( $m$, $r_i$);
end
Sort the sensors by the decreasing of their inscribed circle's radius. $s_{(1)}, s_{(2)}, ..., s_{(n)}$;
for $i$ = 1 to $n$ do
    if $p_{(i)}$!=0 then
        for $j = i + 1$ to $n$ do
            if $r_{(j)} + d(s_{(i)}, s_{(j)}) < r_{(i)}$ then
                $p_{(i)} == 0$;
            end
        end
    end
end
```

Algorithm 1: An algorithm for finding a near minimal blank area given a set of sensor locations $\{s_j\}$ and a desired region R to cover from the inside.

access protected content, and hence should be considered as security weakness. Consequently, we take the viewpoint, that it is desirable to from a security point-of-view to sacrifice some area between the boundary of the STAC region and the actual approximation region so long as we do not have any key information being leaked outside the desired access control region.

To accomplish this, we may adjust the transmission powers to best cover from the inside a particular region R. We now present Algorithm 1, which describes a greedy algorithm for constructing an approximation of a region R from within, with the objective of minimizing the blank area $BA(C,R)$. In this algorithm, the input is the collection of sensor node locations $\{s_j\}$, as well as the desired region R. Additionally, we provide a constraint m which describes the maximum allowed transmission radius for each sensor node. For example, m might be determined by policy or by hardware restrictions. In this algorithm, we assume that there is a direct way to relate the actual transmission power to a corresponding coverage radius p_i (for example, by employing a propagation model). Hence, rather than explicitly define the algorithm in terms of assigning wattage to different nodes, we are instead formulating the problem in an equivalent manner using distances. We use $d(s_i, s_j)$ to denote the distance between the nodes s_i and s_j.

The algorithm basically starts by assigning powers to ensure that the maximal coverage region for each sensor node is as large as possible, while remaining inside of the region R. Then, the algorithm proceeds to remove redundant nodes or power assignments. Using Algorithm 1, it is possible to achieve a more efficient coverage pattern for an arbitrary region R than using a default deployment pattern where every node has the same power assignment. We present an example power configuration that results in Figure 9.

In order to further illustrate the advantages of adapting the power levels, we conducted a simulation study where the

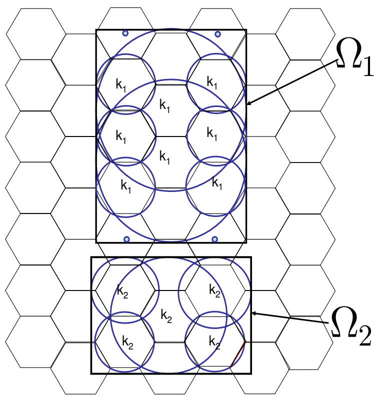

Figure 9: An example of the key distribution coverage pattern after the power allocations of each sensor node have been adjusted using Algorithm 1.

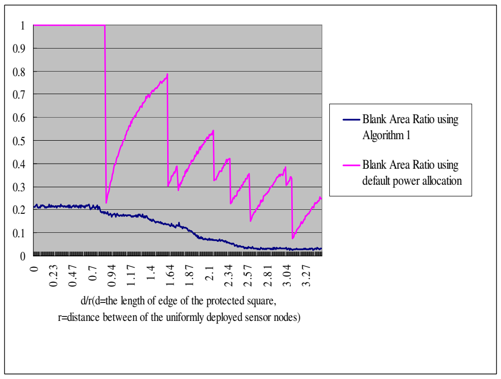

Figure 10: A comparison of the blank area for a uniformly (hexagonal) deployed sensor network with fixed transmit powers with the same network where the transmit powers are adjusted using Algorithm 1. The x-axis corresponds to varying the . The y-axis is the ratio of the blank area to total area of a square with sides d.

ST-region to be protected is a square with sides of length d, and deployed sensors in a uniform hexagonal tiling where the distance between sensor nodes is r. We varied the ratio d/r, and measured the blank area for both the uniform coverage and adaptive coverage resulting from Algorithm 1 using Monte Carlo sampling techniques. We report the results in Figure 10, where the x-axis is parameterized via d/r and the y-axis corresponds to the ratio of the blank area to the area of the total square. In this figure, we see that at small d/r, it is not possible for the uniform deployment to cover the square (going beyond the boundary of the square is not allowed), but the power allocations assigned by Algorithm 1 adapts the transmission power to cover square without going outside the square. As we increase d/r, we allow more sensor nodes to fall within the square, and we see that the adaptive power allocation algorithm consistently results in less blank area than uniform power allocation.

Finally, we note that Algorithm 1 is flexible and can be applied to address the power allocation for any placement of sensor nodes beyond regular lattice patterns.

5.3 Dynamic Encryption and Key Updating

STAC not only involves access based on an entity's spatial location, but also implies that there might be important temporal contexts that affect the ability of a user to access content. There are many cases where a STAC policy for an object (such as an entire movie) might have the requirement that access changes with time. One important example of an object that would have such a policy is a streaming object that varies with time. For this case, it is necessary to decompose an object into smaller object atoms (as defined earlier), and then treat each of these smaller object atoms as individual objects that are protected over a spatial region that is fixed over a smaller time interval. As an example, we can consider an object with a formal ST-region Ω_2, but would have to approximate Ω_2 via a collection of smaller and simpler ST-regions with temporal resolution τ, as depicted in Figure 11. Here, in this figure, region Ω_2 is thus approximated as the union

$$\Omega_2 = \bigcup_{j=1}^{4} \Omega_{2j}.$$

Each of these ST-atoms Ω_{2j} will be enciphered by a corresponding key k_j, and hence the encryption of such an object will be dynamic in the sense that the key will vary with time. For streaming objects, if we were to not employ dynamic encryption, then it would be possible for an adversary to observe the key in a valid ST-region (e.g. Ω_{21}) at a valid time, and then access the rest of the content from a region not allowed by the spatio-temporal access control policy.

As a further issue, we note that dynamic encryption is only meant to protect the content during the initial access period for that content. By this we mean, that once a user has recorded a STAC-protected object and has satisfied the spatio-temporal requirements to access the object, that user has effectively unlocked the object for him/her to access at any later time. Essentially, once a user has the file and the keys, access is granted from that point on. We note that dynamic encryption is only meant to protect dynamically evolving content/objects. If it is desired to strictly limit a user to accessing content during a specific time and not after that time, then it is necessary to employ the use of additional security mechanisms, such as trusted operating systems and secure containers which would guarantee that key information is stored and accessible to the user only during a specified time.

Dynamic encryption and the decomposition of objects into smaller, more refined ST-regions requires that the each ST-region is associated with different keys, and hence the problem of managing the keying information becomes important. Although one could envision that a key distribution center might be able to manage and frequently update keys within the sensor network by issuing updates to each sensor node (e.g. through a gateway between the sensor network and the broader Internet) every time the key needs to change, such a scheme is impractical. Rather, we should reduce the frequency at which the key distribution center interacts with each sensor node by having the center distribute a single set of keys corresponding to a keying schedule.

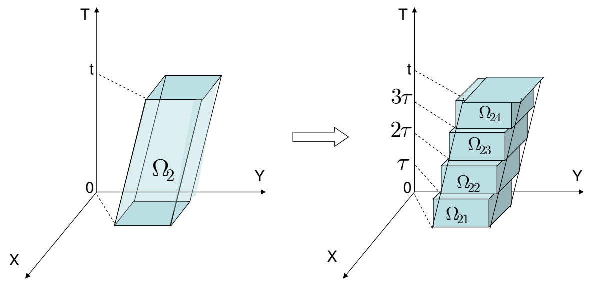

Figure 11: A ST-region Ω_2 is originally specified in a continuous, smooth manner. However, in practice it is necessary to decompose the region into ST region atoms Ω_{2j}, and correspondingly decompose the object into smaller object atoms.

In order to do so, the KDC must either initially install a large set of keys prior to deployment, or the KDC can communicate as needed with the sensor nodes through a set of keys shared between the KDC and each sensor, i.e. $K_{s_i,KDC}$. There is significant overhead associated with frequent updates of keys, and in order to reduce this overhead, we make use of a chain of one-way hash functions to generate and store keys.

A one-way chain $(V_0, ..., V_n)$ is a collection of values such that each value V_i (except the last value V_n) is a one-way function of the next value V_{i+1}. In particular, we have that $V_i = H(V_{i+1})$ for $0 \leq i < N$. Here H is a one-way function, and is often selected as a cryptographic hash function. For setup of the one-way chain, the generator chooses at random the root or seed of the chain, i.e., the value V_n, and derives all previous values V_i by iteratively applying the hash function H as described above, yielding a chain as in:

$$V_0 \leftarrow V_1 \leftarrow V_2 \leftarrow \cdots \leftarrow V_{n-1} \leftarrow V_n.$$

By employing the hash chain, the entity responsible for key distribution need only send the anchor seed and the times at which the sensor node should change keys. For example, if we let the last key be the key seed, i.e. $V_n = K_n$, then the KDC simply performs

$$KDC \rightarrow SN : E_{K_{(s_i,KDC)}}(K_n, t_0, t_1, ...t_n).$$

The sensor then can derive K_1, K_2, all the way up to K_{n-1} locally by applying H. When the keys are used up, the central server will repeat the process.

One necessary system requirement for STAC, though, is that all sensor nodes maintain synchronization with each other and the server so as to guarantee that keys are transmitted during the correct time period.

6. DISCUSSION ON THE OPERATION OF INVERTED SENSOR NETWORKS

The use of the inverted sensor network for spatio-temporal access control achieves several advantages when compared to a centralized scheme: first, it reduces the risk of a privacy breach; second, it is naturally resistant to location spoofing attacks; and third, it facilitates new classes of applications that can easily be implemented.

6.1 Reduced Contextual Privacy Risk

Privacy is the guarantee that information, in its general sense, is observable or decipherable by only those who are intentionally meant to observe or decipher it. The phrase "in its general sense is meant to imply that there may be types of information besides the content of a message that are associated with a message transmission. When you access an ATM at a bank, this action is observable, and some contextual information regarding your actions is revealed to anyone observing you. An adversary that witnesses you going to a bank should naturally conclude that you likely have withdrawn money, and he does not need to launch any sophisticated cryptographic attacks to acquire your money (he simply robs you as you walk away from the bank).

In this bank example, a user's contextual information is revealed. More generally, the issue of contextual privacy has come up in other scenarios, such as database access and location-services. Generally speaking, there is a risk of a privacy breach when any entity A contacts another entity B asking for some service. For example, in the case of database privacy, when an entity A requests information from B, B's sole function should be to provide the service or answer to the query. It might not be desirable for B to know the details of the query or the specific answer [13–15].

In access control systems, there is always the risk of a privacy breach. When the user requests a service, this request can be logged by the entity providing the access control. In Kerberos, for example, all of the emphasis is placed on secure, authenticated exchanges but it is possible for the servers that administer service granting tickets to record which user has made a request for which service, thereby tracking a user's usage patterns or preferences. Similarly, in a centralized spatio-temporal access control, when the user attempts to prove that it is in a specific location, this can be recorded by the centralized entity, and used to infer the user's activities– it becomes possible to not only infer the user's current position, but also by accumulating the user's position over time it is possible to discover the user's habits.

In order to avoid this privacy risk, what is needed is a technique for access control that does not pass through a centralized entity. In our use of inverted sensor networks, the user (or users) are supplied content through some external means. For example, the content may be streaming content that is broadcast, and any entity that wants to access the content is free to do so by simply being at the right place at the right time to acquire the keys transmitted by the inverted sensor network. Since the users don't have to interact explicitly with any entity, there is no information revealed as to whether a specific user is accessing specific content, and there is no information revealed about the user's spatio-temporal profile.

6.2 Resistant to Positioning Spoofing

Any scheme that requires that a user prove that it is in a specific location becomes susceptible to attacks that might be launched against the localization infrastructure [7] (such as an adversary trying to prove that it is in a location that it is not, which would allow the adversary to access content he is not intended to access). Although secure localization schemes can mitigate this threat, there might be a limit to the effectiveness of these techniques against non-cryptographic attacks [8]. By using inverted sensor networks, however, this issue is bypassed. There is no reliance on an entity proving its location to another entity. We do note, however, that inverted sensor networks can be physically attacked by adversaries destroying nodes, capturing and reprogramming nodes, or by simply covering sensors so that their transmissions are blocked. These issues however, are common to all wireless networks, and must be addressed through careful deployment of the sensor nodes (e.g. placing the nodes on the ceiling in a building might make it harder for vandalism).

6.3 Support of Applications with Little Effort

Taking advantage of a sensor network in an inverted fashion in order to facilitate spatio-temporal access control represents what we feel is an exciting opportunity to develop new classes of location-based applications. Programming STAC applications becomes relatively easy with the assistance of an inverted sensor network: sensor nodes can be deployed in support of STAC and then loaded with their key schedule; while a user's application simply needs to receive broadcast content, listen for the appropriate key, and then decrypt the content.

One promising new style of application that we envision is a spatio-temporal scavenger hunt, which might be an interesting paradigm for educational applications. In the scavenger hunt application, the user receives content and can only access it at the right place and right time. This content, once opened, might give the user a puzzle to solve describing where the user must next go to in order to advance to the next stage of the scavenger hunt. If the user solves the puzzle and makes it to the next location within a specified time limit, then the user would get the next puzzle. The process can continue until the user achieves the final objective.

There are many variations that are possible to the basic scavenger hunt. The objective of the scavenger hunt game can be specified by constructing a suitable transition diagram, which would detail the rules and paths allowed for a user to traverse the game. For example, we may create a cut through in the transition diagram (similar to the cut throughs in Figure 5) which would allow for a user to bypass the requirement of going to a certain area and accessing a certain object. This could correspond, for example, to a user solving a more challenging puzzle and being allowed to advance further in the game. Or, as another variation, if a user solves a puzzle and moves to the next location in a short amount of time, then the user might receive a different decryption key than if it had taken longer to solve the puzzle, thus allowing the user to access a different puzzle when it is in this location. Overall, we believe that inverted sensor networks can easily facilitate new classes of applications.

7. RELATED WORK

The conventional literature on access control models can be broken into several main categories: identity-based access control, role-based access control, and context-based access control. Location-based access control may be considered as a specialized form of context-based access control [1, 16].

Research into supporting location-based access control has primarily focused on the issue of providing secure and robust position information. In [17], the authors listed a few attacks that might affect the correctness of localization algorithms along with a few countermeasures. SecRLoc [9] employs a sectored antenna, and presented an algorithm that makes use of the property that two sensor nodes that can hear from each other must be within the distance $2r$ assuming r is fixed in order to defend against attacks. [18] uses hidden and mobile base stations to localize and verify location estimate. Since such base station locations are hard for attackers to infer, it is hard to launch an attack, thereby providing extra security. [19] uses both directional antenna and distance bounding to achieve security. Compared to all these methods, which employ location verification and discard location estimate that indicates under attack, [20] and [8] try to eliminate the effect of attack and still provide good localization. [8] makes use of the data redundancy and robust statistical methods to achieve reliable localization in the presence of attacks.

There has been less work devoted to developing the remaining components needed for spatio-temporal access control, and there has only been a few efforts that have tried to develop location-based access control systems. In [21], the objective is to provide location-based access control to a resource(in this case, the wireless links). The authors propose a Key-independent Wireless Infrastructure (KIWI) where, during the handshake period, KIWI challenges a client who is intended to access the resource with a set of nonces. Only when the client send back the proof that it correctly received all the nonces, can it complete the authentication handshake. In [22, 23], the PAC architecture is proposed in order to provide location-aware access control in pervasive computing environments. Although PAC preserves user anonymity and does not expose a user's exact location, one drawback is that it uses only coarse geographical location areas.

In the area of developing formal access control models, most of the efforts in the literature have focused on general context-sensitive access control models, such as presented in [24]. Here, a formal model is presented which consists of context, policy, request, and algorithm sets. Context information is represented as a six dimensional vector, which includes *time* and *location* as two of the dimensions. This work provides some high-level outlines of authorization and access control protocols that can be based upon their model. Similarly, in [25], the Context Sensitive Access Control framework was presented, which provided a comparison of different access control mechanisms and context verification mechanisms. Two other related efforts were presented in [26] and [27]. In [26], a comprehensive RBAC model is presented that employs location information in its formal model, while [27] employs temporal information in its model.

One important issue related to STAC is user privacy, as techniques that acquire the user's exact location also can expose this information to unwanted parties [11]. Location privacy has recently been studied in the context of location-based services [28, 29]. In [28, 29], a distributed anonymity algorithm was introduced that serves to remove fine levels of detail that could compromise the privacy associated with user locations in location-oriented services. For example, a location-based service might choose to reveal that a group of users is at a specific location (such as an office), or an individual is located in a vague location (such as in a building), but would not reveal that a specific individual is located at a specific location. Duri examined the protection of telematics data by applying a set of privacy and security techniques [30].

8. CONCLUDING REMARKS

As wireless networks become increasingly prevalent, they will provide the means to support new classes of location-based services. One type of location-oriented service that can be deployed are those that make use of spatio-temporal location-information to control access to objects or services. In this paper we have examined the problem of location-based access control by exploring how access control policies can be formally specified, and presenting two different mechanisms for supporting spatio-temporal access control. We first showed how it is possible for a centralized entity, with the assistance of a localization infrastructure, can verify a claimed location. Once a location has been verified, an object can be distributed for access. The second approach that we explored, which was the primary focus of this paper, involved inverting the role of a sensor network. Specifically, an appropriately deployed sensor network can consist of nodes that locally transmit a schedule of keys, thereby facilitating access to enciphered objects. We then examined issues of optimizing the region covered by the sensor network in support of spatio-temporal access control by providing an algorithm for optimizing sensor node power allocation.

We believe that spatio-temporal access control in general, and inverted sensor networks in particular, represent a promising paradigm for the development of new location-oriented applications. The techniques outlined in this paper represent the beginning of a larger effort to develop spatio-temporal applications using inverted sensor networks, and we are currently implementing the system outlined in this paper.

9. REFERENCES

[1] S. Gavrila D. Ferraiolo, R. Sandhu, D. Richard Kuhn, and R. Chandramouli, "Proposed NIST standard for Role-Based Access Control," *ACM Transactions on Information and System Security*, vol. 4, no. 3, pp. 224–274, 2001.

[2] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata theory, languages and computation*, Addison-Wesley Publishing Company, 1979.

[3] P. Bahl and V.N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proceedings of IEEE Infocom 2000*, 2000, pp. 775–784.

[4] P. Bahl, V.N. Padmanabhan, and A. Balachandran, "Enhancements to the RADAR User Location and Tracking System," Tech. Rep. Technical Report MSR-TR-2000-12, Microsoft Research, February 2000.

[5] D. Nicelescu and B. Nath, "DV based positioning in ad hoc networks," *Telecommunication Systems*, vol. 22, no. 1-4, pp. 267–280, 2003.

[6] D. Nicelescu and B. Nath, "Ad hoc positioning (APS) using AOA," in *Proceedings of IEEE Infocom 2003*, 2003, pp. 1734 – 1743.

[7] S. Capkun and J. Hubaux, "Secure Positioning of Wireless Devices with Application to Sensor Networks," in *Proceedings of the IEEE INFOCOM*, 2005, pp. 1917–1928.

[8] Z. Li, W. Trappe, Y. Zhang, and B. Nath, "Robust Statistical Methods for Securing Wireless Localization in Sensor Networks," in *The Fourth International Conference on Information Processing in Sensor Networks (IPSN)*, 2005, pp. 91–98.

[9] L. Lazos and R. Poovendran, "SeRLoc: Secure range-independent localization for wireless sensor networks," in *Proceedings of the 2004 ACM Workshop on Wireless Security*, 2004, pp. 21–30.

[10] N. Sastry, U. Shankar, and D. Wagner, "Secure verification of location claims," in *Proceedings of the 2003 ACM workshop on Wireless security*, 2003, pp. 1–10.

[11] B. Schilit, J. Hong, and M. Gruteser, "Wireless Location Privacy Protection," *Computer*, vol. 36, no. 12, pp. 135–137, 2003.

[12] M. Gruteser and D. Grunwald, "Anonymous Usage of Location-Based Services through Spatial and Temporal Cloaking," in *Proceedings of First ACM/USENIX International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2003, pp. 31–42.

[13] Y. Gertner, S. Goldwasser, and T. Malkin, "A random server model for private information retrieval or how to achieve information theoretic PIR avoiding database replication," *Lecture Notes in Computer Science*, vol. 1518, 1998.

[14] G. D. Crescenzo, Y. Ishai, and R. Ostrovsky, "Universal service-providers for database private information retrieval (extended abstract)," in *Symposium on Principles of Distributed Computing*, 1998, pp. 91–100.

[15] C. Cachin, S. Micali, and M. Stadler, "Computationally private information retrieval with polylogarithmic communication," *Lecture Notes in Computer Science*, vol. 1592, 1999.

[16] M. Bishop, *Computer Security: Art and Practice*, Addison Wesley, 2003.

[17] S. Capkun and J.P. Hubaux, "Secure positioning in sensor networks," Technical report EPFL/IC/200444, May 2004.

[18] S. Capkun and J.P. Hubaux, "Securing localization with hidden and mobile base stations," Proceedings of IEEE Infocom 2006.

[19] L. Lazos, R. Poovendran, and S. Capkun, "Rope: robust position estimation in wireless sensor networks," in *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks (IPSN 2005)*, 2005, pp. 324–331.

[20] D. Liu, P. Ning, and W. Du, "Attack-resistant location estimation in sensor networks," in *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks (IPSN 2005)*, 2005.

[21] D. B. Faria and D. R. Cheriton, "No Longterm Secrets: Location-based Security in Overprovisioned Wireless LANs," in *Proceedings of the Third ACM Workshop on Hot Topics in Networks*, 2004.

[22] N. Michalakis, "PAC: Location Aware Access Control for Pervasive Computing Environments," 16 September 2002.

[23] Nikolaos Michalakis, "Location-aware Access Control for Pervasive Computing Environments," .

[24] W. Han, J. Zhang, and X. Yao, "Context-sensitive access control model and implementation," in *The Fifth International Conference on Computer and Information Technology*, pp. 757–763.

[25] R. J. Hulsebosch, A. H. Salden, M. S. Bargh, P. W. G. Ebben, and J. Reitsma, "Context sensitive access control," in *SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies*, New York, NY, USA, 2005, pp. 111–119, ACM Press.

[26] E. Bertino, B. Catania, M. L. Damiani, and P. Perlasca, "GEO-RBAC: a spatially aware RBAC," in *SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies*, New York, NY, USA, 2005, pp. 29–37, ACM Press.

[27] J. Joshi, E. Bertino, U. Latif, and A. Ghafoor, "A generalized temporal role-based access control model," vol. 17, pp. 4–23, 2005.

[28] M. Gruteser, G. Schelle, A. Jain, R. Han, and D. Grunwald, "Privacy-aware location sensor networks," in *Workshop on Hot Topics in Operating Systems (HotOS)*, 2003.

[29] M. Gruteser and D. Grunwald, "Anonymous Usage of Location-based Services through Spatial and Temporal Cloaking," in *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2003.

[30] S. Duri, M. Gruteser, X. Liu, P. Moskowitz, R. Perez, M. Singh, and J. Tang, "Context and Location: Framework for security and privacy in automotive telematics," in *Proceedings of the 2nd international workshop on Mobile commerce*, 2002.