Remote Monitoring Architectures As Part of the Frontier

Philip L. Campbell, Richard L. Craft, Lillian A. Snyder
Sandia National Laboratories
Albuquerque, New Mexico 87175
{plcampb, rlcraft, lasnyde}@sandia.gov

Abstract: This paper presents a characterization of "remote monitoring architectures," such as those used for treaty verification or the monitoring of hazardous materials. These architectures are on the forefront of technology. The solutions they will generate will become commonplace features in the future. For example, these architectures generally assume that the adversary is a legitimate user. In this paper an abstract partitioning of the set of architectures is presented, along with brief presentations of several examples.

1 Introduction

Remote monitoring architectures operate in a high risk environment over which the designer does not have complete control and within which the adversary may be a legitimate user. Industry is moving toward a similar environment. The explosive growth of the Internet is the latest evidence that users want to use this type of environment, even if it means dubious security. We believe that this trend will continue, simply because it represents a gold mine of opportunity [1]. As a result the solutions that remote monitoring architectures will develop will become commonplace in systems of the future.

Remote monitoring architectures have been of interest to Sandia National Laboratories for several decades. The application of interest has been the development of a treaty verification system in

which a host country allows the monitor, who is also the host's adversary, to place seismometers on its (the host's) soil. The monitor wants to be guaranteed that it receives all of the seismic data. The host wants to be guaranteed that only the monitor receives only the seismic data—eavesdropping and covert channels must be thwarted. It would appear that these requirements are "mutually exclusive and irreconcilable" [3]. A "compromise solution" was developed in the early 1970's—a "digital data authenticator" [4] as it was called then or message authenticating code (MAC) as it would be called today. Under this scheme the seismometric device would attach to the raw data a MAC that could include a unique message identifier such as a message number. The host could then view the data, satisfying the requirement that the data is legitimate seismic data, after which the host would encrypt both the raw data and the MAC using a symmetric key encryption algorithm—the only kind available at that time. This scheme satisfies all of the requirements except for the host's stipulation that covert channels be eliminated. The problem is that the host can not verify the MAC without receiving the key that would simultaneously allow the host to cheat by changing the raw data and producing a new MAC to match it. This part of the problem was addressed by the advent of asymmetric encryption systems: the host could verify the MAC without being able to produce a legitimate one. Asymmetric encryption also provided a way to provide additional services: it enabled both parties to be able to convince a third-party of the other's non-compliance and/or their own compliance. As additional services have been required and new application areas investigated, the set of these architectures has grown.

The rest of this paper is organized as follows. We first present an abstract definition of remote monitoring architectures. We begin with an abstract model, then develop partitions based on variations in that model and on combinations of secure services and data sensitivity. We then present

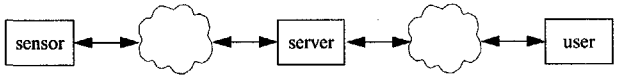
DISCLAIMER

several architectures, most of them general architectures, and show where they fit in our partitions.

2 Defining Remote Monitoring Architectures

The Sjulin-Moore Model,[1] shown in Figure 1, presents an abstract view of remote monitoring architectures.

Figure 1. Sjulin-Moore Model



The clouds represent communication paths open to adversarial attack.

The sensor, server, and user themselves are assumed to be protected from adversarial attack.

The server has storage but the sensor does not.

This model allows the following variations: the server may be absent; the communication paths may be unidirectional; and one or both of the clouds may be absent—the absence of a cloud indicates that the corresponding communication line is protected from adversarial attack. For ease of discussion, the information flowing toward the user—from left to right in Figure 1—is referred to as sensor data; the information flowing toward the sensor—from right to left in Figure 1—is referred to as commands.

1. This model is named after its developers, Mike Sjulin and Judy Moore of Sandia National Labs. It is an abstraction of the "Notional Remote Monitoring System" [2].

The Sjulin-Moore model enables us to grasp the set of remote monitoring architectures but, as with any abstraction, it does so at the cost of simplicity. The Model ignores at least the following: multiple sensors with different generative rates, resolution, power consumption, and command capabilities, aggregated in different ways, operating in different locations under different jurisdictions; multiple servers with different functionality—most likely distributed—with different storage capabilities, characteristics, and inter-server communication paths; different users and types of users with different combinations of demands and constraints; different communication paths between all parts of the system (e.g., direct sensor-to-user communication in a system with a server).

In this section we present the general features of all remote monitoring architectures, then we present the user population, the typical application areas, and the secure services associated with these architectures.

2.1 General features

Remote monitoring architectures share the following features:

1. the system consists of three general parts: a sensor, a server, and a user, with communication lines connecting sensor and server and connecting server and user, as shown in the Sjulin-Moore model (see Figure 1);
2. the sensor device generates data of interest to the user;
3. the server may or may not be present, but if present, it sits logically between the sensor and the user;

4. there is a host that may be the user; if the host is not the user, then the host may be an adversary of the user;

5. the sensor is on the host's soil;

6. the communication lines between sensor, server, and user may be outside the control of the host and user;

7. non-repudiation may be of great interest;

8. there may be many sensors and/or many servers and/or many users.

2.2 Application Areas

There are three general application areas of remote monitoring architectures. The architecture can monitor an object (e.g., a nuclear weapon, an item of nuclear material, a facility), a process, or an activity.

An example of an object is the storage of weapons-grade material or actual nuclear weapons. An example of a process is the mixing, under treaty, of weapons-grade uranium with non-weapons-grade uranium to form non-weapons-grade uranium. An example of an activity is seismic activity, particularly seismic activity that would reveal the explosion of nuclear devices. In all of the examples given here the stakes are high and the adversary is intensely interested in cheating.

2.3 User population

The users of remote monitoring architectures divide into four groups:

1. host—the owner of the soil on which the sensor is placed;

2. interested party—an organization that has no power over the host and is not in a treaty agreement but for whom the host is willing to provide access to certain information, perhaps to show that the host is a good world-citizen;

3. monitoring organization—an organization that is officially recognized internationally and thus has at least the power of world opinion;

4. treaty partner—a nation that has entered into a treaty with the host.

Note that in the case of the treaty partner that the adversary is not an unidentified rogue whose presence may not even be detected. Rather, the adversary is a legitimate user of the system! For some nations, a monitoring organization may be considered to be in the same category.

2.4 Data Sensitivity

We presume that all information in the system is assigned one of two sensitivity levels to which we will give the names "classified" and "unclassified." The names we have chosen are secondary to the characteristics of the associated information. Classified information is always protected at least as well as unclassified. And unclassified information can be shared with at least as many people as classified.

In the current application areas for remote monitoring architectures classified information is highly protected—always—and never shared with individuals that have not received a "clearance." We presume that no foreign nationals would ever have clearances and thus classified information would never be shared with them. Generally we presume that it is acceptable to share unclassified information with a broader population base than classified information, but restrictions would almost always still apply.

2.5 Secure Services

We presume that authenticity is required on all communication lines because none of the parties may be able to corroborate the information flow via another source. Also, the information may lose all value if it is not authenticated. For example, seismometer data is essentially worthless without knowing the location of the seismometer.

In addition, we presume that freedom-from-inference is required on all communication lines. This security service applies to information channels for which the possible data values are known and the actual data sent corresponds to actions made by the receiver. An eavesdropper can thus make valuable inferences. For example, suppose that Country X is interested in seismographic information generated by sensors on Country Y's soil, but Country X receives no data unless it requests it. Suppose further that this system is shared with several other countries. Now, it is likely that neither the commands (i.e., the requests) nor the answering data is particularly sensitive—knowing a command by itself or a piece of sensor data by itself may be available to all of the participating countries. However, knowing that Country X issued a given command at a particular moment—or being able to infer the command given the data that Country X has received—would probably be highly sensitive since it implies Country X's interests and its suspicions. It may also reveal patterns that Country Y could use to enable it to generate extra-treaty seismographic pulses that elude Country X's attention. One way to provide this service is to provide Country X with sufficient additional, unrequested information that an eavesdropper would be unable to infer Country X's interests. Another way to provide this service is to provide confidentiality—that is, confidentiality subsumes freedom-from-inference.

As noted in Section 2.1, the users (and the host) of remote monitoring architectures are interested in non-repudiation services: proof-of-origin, proof-of-submission, and proof-of-receipt. Proof-of-origin is a receiver service; the other two are sender services.

We presume the following constraints.

1. Classified information must always be provided confidentiality, due to the nature of classified information.
2. Proof-of-receipt subsumes proof-of-submission: if you have the former, you do not need the latter.
3. Classified information is never combined with any of the non-repudiation services. A host would never share classified information with anyone else, and non-repudiation services are never needed when the host is the user.
4. *Confidentiality is provided by default.*

Applying the above constraints to the 32 possible combinations of secure services and data sensitivity reduces them to eight, as shown in Table 1.

Table 1. Combinations of Secure Services & Data Sensitivity[a]

	data classified?	confidentiality?	proof-of-		
			-origin?	-submission?	-receipt?
1	no	no	no	no	no
2	no	yes	no	no	no
3	yes	yes	no	no	no

Table 1. Combinations of Secure Services & Data Sensitivity[a]

	data classified?	confidentiality?	proof-of-		
			-origin?	-submission?	-receipt?
4	no	yes	yes	no	no
5	no	yes	no	yes	no
6	no	yes	no	no	yes
7	no	yes	yes	yes	no
8	no	yes	yes	no	yes

a. *Authenticity and freedom-from-inference is assumed in all cases. Note that proof-of-origin provides* authenticity and that confidentiality provides freedom-from-inference.

3 Partitioning the Remote Monitoring Architectures

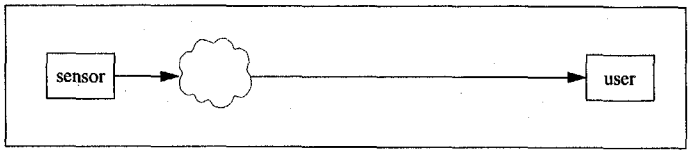
We can deepen our understanding of remote monitoring architectures by taking a closer at the terrain they define. We can partition this terrain using the Sjulin-Moore model, and then we can partition each of those partitions by using secure services and data sensitivity.

3.1 Using the Sjulin-Moore Model as the Basis

The Sjulin-Moore model provides a partitioning scheme based on the presence/absence of the server and the directionality of the remaining communication lines. This approach results in six partitions: Push, Pull, Push-Push, Push-Pull, Pull-Push, and Pull-Pull. The term "pull" refers to the presence of commands; the term "push" refers to the absence of commands. The diagram for each partition is shown in a Figure below.

The diagram for the systems in the Push partition is shown in Figure 2.

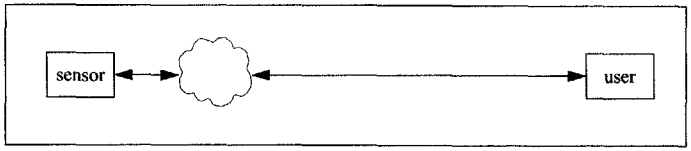
Figure 2. (1) The Push Partition



There is no server in these systems and the user is unable to issue commands. Since there is no information flow to the sensor, these systems can provide a higher level of security than any of the Pull systems can. It is possible for different users to be connected to different sensors, but every user that is connected to a given sensor is sent all of the information in real-time that is generated by that sensor.

The diagram for the systems in the Pull partition is shown in Figure 3

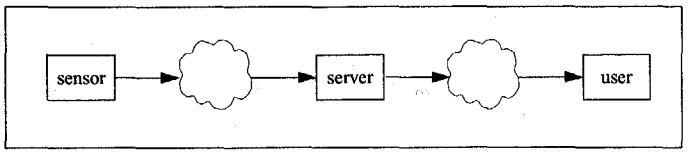
Figure 3. (2) The Pull Partition



Like the systems in the Push partition, the systems in the Pull partition have no server. However, the user is able to issue commands. Again, like the systems in the Push partition, it is possible for different users to be connected to different sensors, and every user that is connected to a given sensor is sent all of the information in real-time that is sent by that sensor.

The diagram for the systems in the Push-Push partition is shown in Figure 4.

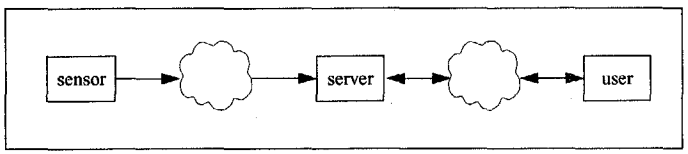
Figure 4. (3) The Push-Push Partition



The systems in this partition have a server but no commands can be issued. The presence of the server implies that there is storage available—the sensor data is no longer required to flow to the user in real-time as it was in the Push and Pull partitions. The server may provide the raw data; it may condense it; it may change its classification; it may massage it. There may be multiple servers and they may be connected serially, making it possible for the same user to receive the same sensor data in raw, condensed, differently-classified, and massaged form. But note that the user cannot direct the server and the server cannot direct the sensor.

The diagram for the systems in the Push-Pull partition is shown in Figure 5.

Figure 5. (4) The Push-Pull Partition

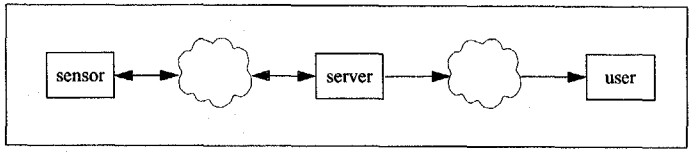


The systems in this partition also have a server, but the user can issue commands. This enables the user to direct the distribution of raw data and/or direct the massaging of data. The diagram does not specify the extent of the control that the user has over the server—it may be minuscule. But

note that since the server cannot issue commands, neither the server nor the user can direct any of the activity of the sensor.

The diagram for the systems in the Pull-Push partition is shown in Figure 6.

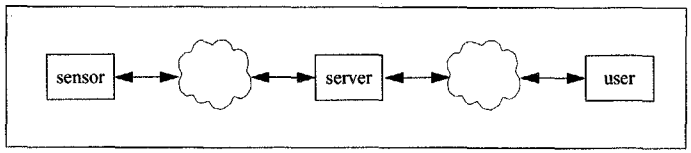
Figure 6. (5) The Pull-Push Partition



The systems in this partition allow the server to direct the activities of the sensor. As with the Push-Pull diagram, this diagram does not specify the extent of the control that the server has over the sensor. It is possible that the server could ask for changes in the sensors based on the sensor data itself. For example, if the sensor data suggested that a seismic event were occurring, the server could ask for an increase in sensing frequency for the sensors in the interesting area. Additionally, the server could direct cameras toward a door whose alarm has just been triggered.

The diagram for the systems in the Pull-Pull partition is shown in Figure 7.

Figure 7. (6) The Pull-Pull Partition



The systems in this final partition allow the server to direct the activities of the sensor and also allow the user to direct the activities of the server, all at least potentially. It may be that the server

could pass user commands on through to the sensor. But on the other hand the server could filter such commands or disallow them entirely, letting only its own commands be issued to the sensor.

3.2 A Second Partition: Using Secure Services & Data Sensitivity

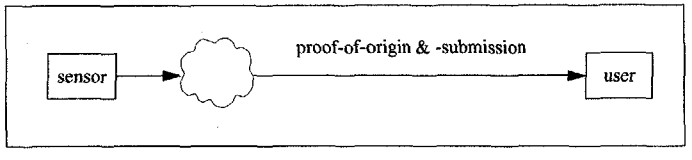
We further partition the Push/Pull partitions presented in Section 3.1 by using the combinations of secure services and data sensitivity presented in Section 2.5. Since there are eight possible combinations, the number of representative systems in each partitions depends on the number of available, uni-directional communication lines, as shown in Table 2.

Table 2. Partition Sizes

Partition	Communication Lines	Partition Size
Push	1	$8^1 = 8$
Pull	2	$8^2 = 64$
Push-Push		
Push-Pull	3	$8^3 = 512$
Pull-Push		
Pull-Pull	4	$8^4 = 4{,}196$

As an example of one of the eight systems in the Push partition, consider Figure 8.

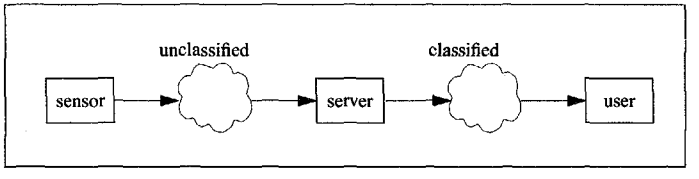
Figure 8. A Push System



The communication line to the user provides proof-of-origin to the user and proof-of-submission to the sensor. Since proof-of-origin is provided we can conclude that the user is probably a monitoring organization or a treaty partner but not the host or an interested party. Since the sensor is provided proof-of-submission we can conclude that the sensor is interested in being able to prove to a third-party that the data was in fact sent. But note that the sensor is not provided with proof-of-receipt. If given the choice, we presume that the sensor would want proof-of-receipt since this is stronger. But the fact that the sensor is not provided this service implies some constraint that precluded it. Perhaps the service would cost too much. Perhaps it would be politically unwise to provide the service: with proof-of-submission the sensor can be free of blame without simultaneously showing negligence on the part of the user—noise on the communication line can always be blamed.

As an example of one of the 64 systems in the Push partition, consider Figure 9.

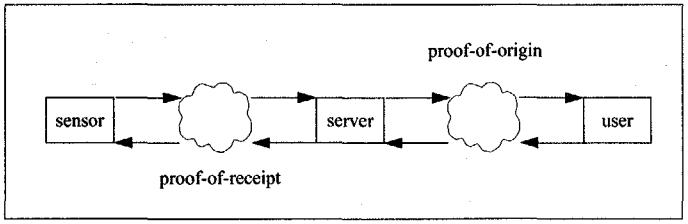
Figure 9. A Push-Push System



The communication line delivers classified to the user so the host would not allow anyone else to use this system. The data from the sensors is unclassified, so the server must be condensing or correlating or massaging the data in some way that it becomes classified.

As an example of one of the four thousand systems in the Pull-Pull partition, consider Figure 10.

Figure 10. A Pull-Pull System



The user is provided proof-of-origin from the server but not from the sensor. We can conclude from this that the user trusts the sensor but not the server. The server is not provided non-repudiation in its communication to the user. We can conclude from this that the server is not held accountable for not sending data. Meanwhile, the server is provided proof-of-receipt to the sensor. We can conclude from this that the server does not trust the sensor. The sensor and server do not belong to the same party, neither do the server and user.

4 Sample Architectures

In this section we present high-level views of several remote monitoring architectures. The purpose here is to provide the reader with an understanding of actual architectures. We present views of the architecture for the Self-Aware Weapon / Information Infrastructure (SAW/II), StraightLine, Modular Integrated Monitoring System (MIMS), the International Monitoring System for the Comprehensive Test Ban Treaty (CTBT IMS), the Integrated Intrusion Detection and Access Control Annunciator (IDACA), and an architecture for monitoring fluid mix—this last architecture has no official name. Since most of these architectures are general designs, most of them fall

into the most general partition, Pull-Pull. Table 3 categorizes the architectures based on the partitions presented in Section 3.

Table 3. Summary of Sample Architectures

Architecture	Push/Pull	Data Sensitivity	User
SAW/II	Push-Pull	classified	(all but treaty partner; the classified data would be declassified if the user were a monitoring organization)
StraightLine	Push-Pull or Pull-Pull, depending on the implementation	classified	(all)
MIMS	Pull-Pull	unclassified?	Monitoring organization
CTBT IMS	Pull-Pull	unclassified	Treaty partner
IDACA	Pull-Pull	unclassified	Host
Monitoring fluid mix	Push	unclassified	Treaty partner

The architecture for SAW/II is shown in three Figures, Figure 11, Figure 12, and Figure 13, each subsequent Figure expanding on the previous one.

Figure 11. SAW/II System Architecture

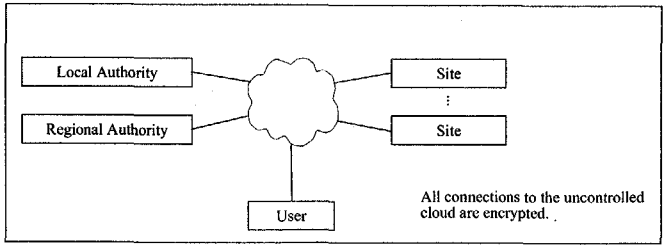


Figure 12. SAW/II Site Architecture

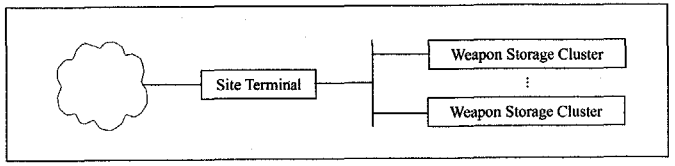
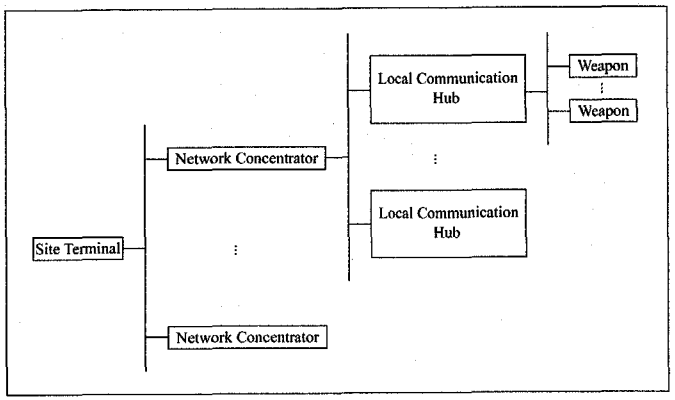


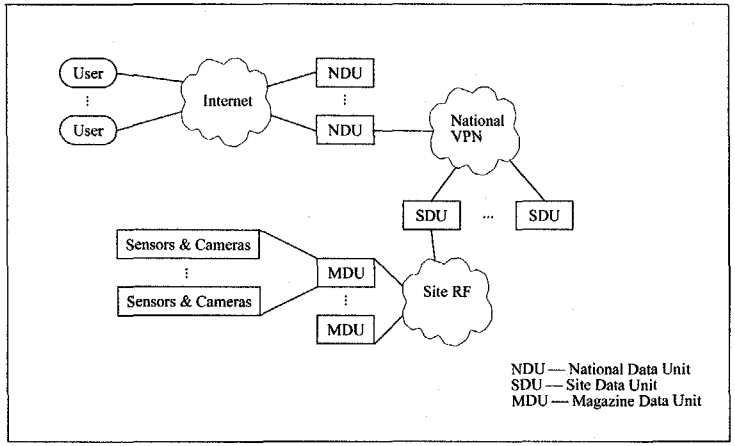
Figure 13. SAW/II Weapon Storage Cluster



SAW/II is intended to be used with nuclear weapons. It is assumed that the weapons are geographically dispersed and that the sensing devices on the weapons themselves are expected to operate for long periods of time on battery power alone. The architecture presumes that the sensing devices push the data to the servers. This has a security aspect to it, besides the practicality of reducing the drain on precious power. The large number of layers in the architecture reflects the expected geographical distribution along with the need to be able to process commands away from the sensors as well as the expected highly diverse user population.

The StraightLine architecture is shown in Figure 14.

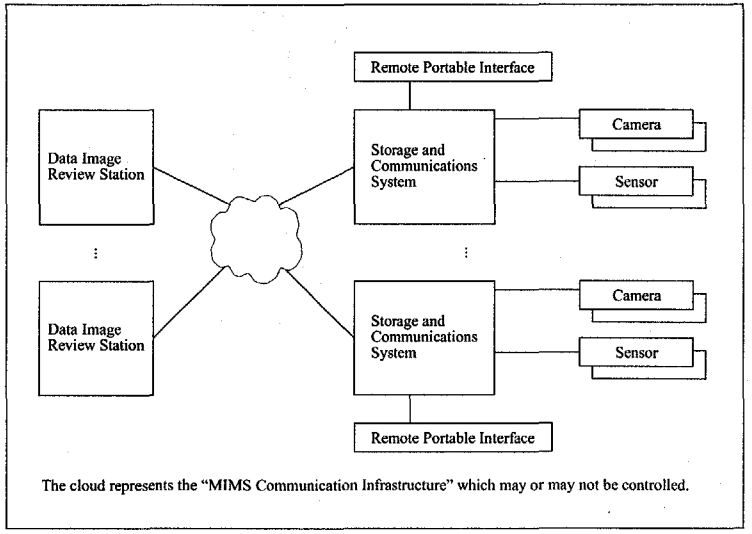
Figure 14. StraightLine Architecture



StraightLine is intended to monitor nuclear materials. The data from the sensors and cameras funnels to the National Data Unit (NDU). Each participating country would provide its own NDU. A country's NDU is the server for the queries concerning that country's nuclear materials. StraightLine is a Push-Pull or Pull-Pull system, depending on the implementation. One alternate design split the communication line between the SDUs and the NDUs, sending classified information on one network and unclassified information on another. It is probable that commands from the classified network would not be allowed to proceed back to the sensor so this would be a Push-Pull system. We would expect a wide variation in the filtering of commands from the user to the sensors & cameras.

The MIMS architecture is shown in Figure 15.

Figure 15. MIMS System Architecture



MIMS is designed to be a generic monitoring architecture for the use of the International Atomic Energy Agency (IAEA) to monitor materials or processes. An interesting twist that this architecture provides is the Remote Portable Interface, which allows the user to circumvent the possibly uncontrolled communication lines between the server and the user.

The CTBT IMS architecture is shown in three figures, Figure 16, Figure 17, and Figure 18.

Figure 16. Logical Structure of CTBT IMS

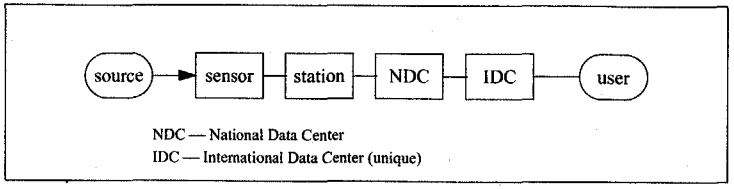


Figure 17. CTBT IMS Architecture

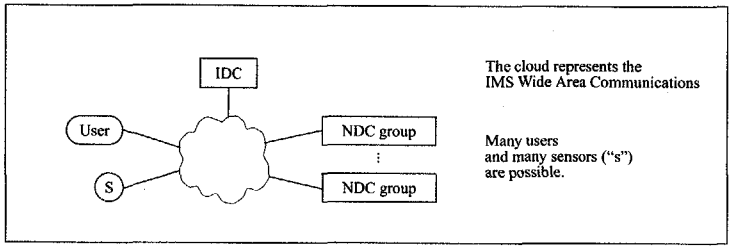
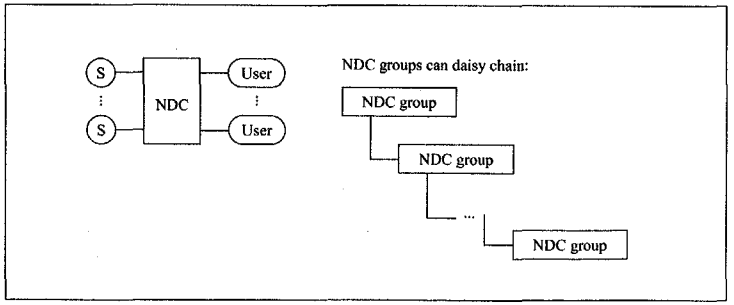


Figure 18. National Data Center (NDC) group

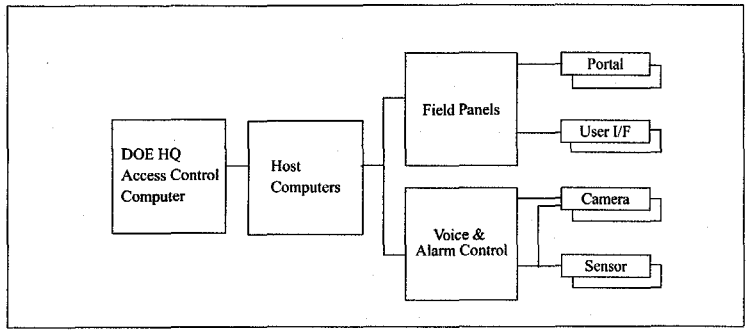


The logical structure of CTBT IMS is similar to StraightLine except that CTBT IMS has a single hub, the IDC. This single point is intended to control the flow of all data and commands. As we

would expect, the architecture does not rule out the possibilities for users to receive data from sensors in their own countries without going through the IDC. (??? when done, pursue getting it so that new pages are always Right master page; look at column layout, I think it is)

The IDACA architecture is shown in Figure 19.

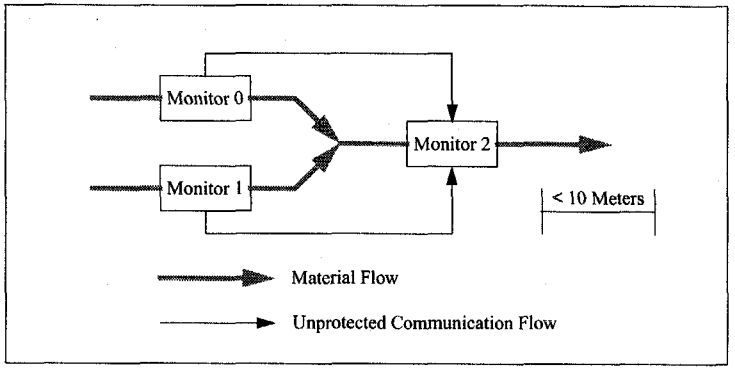
Figure 19. IDACA Architecture



IDACA was designed for monitoring sites within the U.S. Department of Energy. The variety of sensing devices indicates the variety of activities that this architecture is intended to monitor. There would be one set of "Host Computers" at each site. These machines would be responsible for alerting security personnel of problems. The HQ computer would provide information between sites.

An architecture for monitoring material fluid mix is shown in Figure 20.

Figure 20. Material fluid mix



The scale indicates correctly that this architecture is designed to be used within a single building, presumably on the adversary's soil, within a secured compound also controlled by the adversary. The user is an inspector that is provided periodic on-site access to the output of Monitor 2 and monitored inspection of the other Monitors to check for component "failures" (i.e., attempts by the adversary to subvert the tamper-indicating enclosures that surround the Monitors).

5 Conclusions

We have presented the set of remote monitoring architectures by developing partitions based on an abstract model of the architectures. We provided more detail in each partition by developing second-level partitions based on secure services and data sensitivity. We then presented several remote monitoring architectures. The combination of partitions of abstract architectures and actual ones provides an understanding of these architectures. We believe that the solutions pro-

vided by these architectures will become increasingly important as the industry in general moves to an environment that is similar to that of the remote monitoring architectures.

References

[1] P. L. Campbell, R. L. Craft, L. A. Snyder, "Secure Architectures: A Look to the Future." (in preparation).

[2] R. L. Craft, "A Somewhat Definitive Guide to the Potential Future Needs of Information Surety at Sandia National Labs," November 1996.

[3] G. J. Simmons, "How to Insure that Data Acquired to Verify Treaty Compliance Are Trustworthy." Proceedings of the IEEE, vol. 76, no. 5. May 1988. (Reprinted as Chapter 13 of "Contemporary Cryptology," edited by G. J. Simmons, IEEE Press, 1992.)

[4] G. J. Simmons, R. E. D. Stewart, P. A. Stokes, "Digital data authenticator." Patent Application SD2654, S42640, June 30, 1972.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.