

AN MSLS-EMM FOR ENFORCING CONFIDENTIALITY IN MALICIOUS ENVIRONMENTS

Bei Wang and Jim Alves-Foss
Center for Secure and Dependable Systems
University of Idaho
POBOX 441008
Moscow ID 83844, USA
email: [wang4056, jimaf]@uidaho.edu

ABSTRACT

The use of security policy enforcement mechanisms has been a topic in recent literature. Particular focus has been on the class of policies that can be enforced by these mechanisms but not on the security policy guiding the execution of the monitoring mechanisms. It has been a challenge to enforce information confidentiality in a multi-level secure system since malicious users can exploit covert channels within the enforcement mechanisms to propagate confidential information. In this paper, we characterize necessary security properties for an enforcement mechanism that can ensure secure execution of the untrusted programs even though they may be malicious.

KEY WORDS

execution monitoring mechanisms, confidentiality, multi-level secure,

1. Introduction

The security of a computer system relies on two important things: a sound security policy that rules out unacceptable behaviors and a trustworthy enforcement mechanism that guarantees the policy is respected. Recently Schneider \cite{Sch00} characterized a class of security policies that can be enforced by monitoring a target system's execution steps; the corresponding enforcement mechanism is named *Execution Monitoring Mechanism* (EMM). An EMM ensures the secure execution of untrusted programs. Example EMMs include MILS Guard, MILS Message Router [1, 2], and firewalls. Additional required properties of EMM enforceable policies have been further identified [3, 4, 11].

While the research on EMM enforceable policies is abundant, the security of an EMM has been ignored. An EMM is usually deployed in a target system whose behaviors are not trusted to conform to the desired security policy. These untrusted programs can utilize the deficiency or shared resources of an EMM to violate the security policy. The main goal of this paper is to build a security model for an EMM that is robust to such attacks in a multi-level secure system. By security we mean 1) an

EMM faithfully enforces security policies and 2) the deployment of an EMM will not compromise the security of the system it enforces. In other words, the composition of an EMM with a target system will maintain the security of the system. From the perspective of an EMM, there are two types of composition. First is the composition of EMMs and second is the composition of an EMM with a system it monitors. The former requires that the security property be composable with itself while the latter demands the security property preserve other components' (in particular that of the target system) security property. This implies that the security property of an EMM must be stronger than the security property of the system it monitors [7]. An ideal secure model must exhibit both composition properties.

The necessary security property of an enforcement mechanism cannot be determined without taking the security policy it enforces into consideration [12]. We focus on investigating the security property of a class of EMMs that enforce confidentiality and thus the necessary security constraints that prevent confidential information leakage of a target system through observation of or collaboration with EMMs' behaviors.

The paper is organized as follows. We identify the possible information flow paths within an EMM and resulting exploitation scenarios in Section 2. The necessary security requirements of an EMM are summarized in Section 3 and an existent security model is applied to formalize the requirements. In Section 4, an example security policy is introduced to demonstrate the appropriateness of the security model. The paper concludes in Section 5.

2. Information Flow Analysis

An EMM enforces a security policy through intercepting a target system's execution steps and preventing illegal behaviors from occurring. The target system is secure if the composition of an EMM(s) and the target system is secure. In order to facilitate the analysis of EMM security, we call the system an EMM monitors the *environment*. We distinguish two types of information flow: *EMM information flow* and *environment information flow*.

EMM information flow is the flow that occurs within an EMM and can be observed through its input/output sequences while environment information flow is the flow that occurs in an EMM's environment and is visible by pertinent environment components. It is the EMM information flow that is subject to the regulation of an EMM's security constraints.

2.1 Identifying EMM Information Flow Paths

One of the difficulties of controlling confidential information flow in a computer system is due to the existence of covert communication channels. A *covert channel* is a mechanism that utilizes shared resources between users with different security levels to leak information. We therefore separate two types of information flow paths in an EMM, *overt information flow paths* and *covert information flow paths*. An overt information flow path provides a direct access method from the information container to the receiver; a covert information flow path allows the information container to interfere with shared system resources to disclose information. The security policy of an EMM should eliminate the undesired information flow through both paths. In order to identify covert channels in an EMM, it is necessary to identify the shared resources that can be exploited to leak information.

Sharing storage space: All the untrusted programs share the input and output buffers of an EMM, so there may exist covert storage channels. A covert channel is a storage channel if a malicious observer can receive different responses for the same input within a fixed time.

Sharing EMM computation capability: We define an EMM computation capability C_{emm} as the number of messages an EMM can process in a unit of time. A response time R of an input message is the time interval from the time a message is sent to an EMM to the time an output is received. A computation time T of a message is the time an EMM uses to compute the response with respect to the security policy. We assume these parameters are known to all the untrusted programs in a system. Since the untrusted programs share the computation capability of the EMM, the response time of each message varies depending on the pending message in the buffer. A covert channel is a timing channel if a malicious observer can receive the same response for the same input in varying response time R .

Reclassification: We define the change of the security level of an entity as a type of covert channel in that 1) the security levels are shared resources of the untrusted programs, and 2) reclassifying an entity could interfere with the observation of other entities classified at the entity's former level. Reclassification includes both upgrading and downgrading of a message. A message is upgraded if it is reassigned a higher security label, while a message is downgraded if it is reassigned a lower security label.

The problem with downgrading is that it leads to direct information flow from a high level entity to a low

level entity. Upgrading could result in undesired information flow as well, if upgrading is manipulatable by a malicious entity. An example covert channel is illustrated in Figure 1 where in_i/out_i and in_j/out_j represent inputs and outputs classified at security levels i and j , respectively, and $i > j$. The dashed line separates security levels. The observation of the user at level j in Figure 1(a) is $\langle in_j, out_j \rangle$, while the observation of the user at level j in Figure 1(b) becomes in_j . The output event out'_j becomes invisible for the user at level j . This loss of observation of low level events can be exploited by high and low entities to leak information. The bandwidth of such a channel depends on the speed of the generation of low level events and time needed for upgrading information.

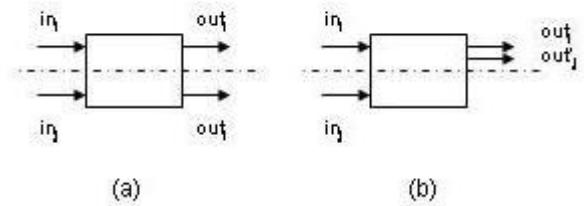


Figure 1. An Example covert channel resulting from upgrading.

2.2 Exploitation of Information Flow Paths

In this section we present the possible exploitations of the identified paths. To simplify the presentation of exploitations of the information flow paths, we assume that the environment of an EMM consists of a set of system components which only have two security levels, H and L. These system components are untrusted programs and may behave maliciously. The interface of an EMM is divided into two levels, H and L, as well. More specifically, H is a set of high level inputs generated by H system components and a set of high level outputs, which can be observed only by high level users and we wish to keep secret from low level users, while L is a set of low level input created by L system components and a set of low level outputs, which can be read by all system components. An information flow path is a communication mechanism that enables H to interfere with L's events.

We assume all L users have *a priori* knowledge of the specification of the system and thus know all possible outputs from H that are compatible with L's inputs. This knowledge enables L to detect information from H based on the outputs it received. Since there are only two users in our system, L can clearly determine interference coming from H; the covert channels we present here are noiseless. Noise is the amount of information that increases the uncertainty of the deduced information, e.g., an interference from a user *chaos*. A noiseless covert channel represents the upper bound of attack scenarios. Formalization and classification of noisy covert channels are being conducted in a related project [10].

In the remainder of the section we summarize six possible exploitation scenarios. We assume that both an EMM and its environment components can be malicious. More specifically, in Scenarios 1-5 we assume an EMM is malicious, and in Scenario 6 only H or L may be malicious.

Exploitation Scenario 1:

1. EMM stores H's information.
 2. L sends a message to the EMM to request the information.
- EMM copies H's information to L's message.

In this scenario, a malicious EMM duplicates H's information and transfers it to L's output channel, through which L can observe the information. This is a violation of overt information flow: a high level process writes information to a low level entity.

Exploitation Scenario 2:

1. EMM receives H's message.
2. It downgrades the message and sends it through L's output channel.
3. L obtains the entire information of the message if it is in plaintext and one bit of information if it is in ciphertext, since the existence of the message encodes one bit of information.

Information leakage due to downgrading is much more complicated than the scenario example presented here. The difficulties arise from the transitive nature of the information flow model, in which information flow is stipulated by a lattice while transitivity is inherent in lattices. An intransitive information flow model is required to control downgrading. For more information about downgrading, please refer to work by Zdancewic [14].

Exploitation Scenario 3:

1. L sends a message to the EMM to request H's information.
2. EMM retrieves the information from its memory and encodes the information using predefined strategies.
3. L generates the next input. EMM upgrades the input and transmits it through H's output channel. L loses the view of its response and interprets it as one bit of information.
4. L continually generates the same input message, EMM strategically upgrades or passes through the message to disclose H's information to L.

A malicious EMM utilizes its upgrading privilege to leak information. L repeatedly sends the same request to the EMM and waits for the response. It can deduce one bit of information from receiving or not receiving an output message.

Exploitation Scenario 4:

1. L sends a message to EMM to request H's information.
2. EMM retrieves the information from its memory and encodes the information using predefined strategies.
3. EMM rejects L's next input to transmit a '0' and accepts L's next input to transmit a '1'.
4. L continually generates the same input to the EMM and step 3 repeats until transmission completes.

This is an example of exploitation of the non-input total¹ feature [13]. An EMM controls the acceptance or rejection of an input, which can be utilized to leak H's information to the sender of a request, L in this case.

Exploitation Scenario 5:

1. L sends a message to EMM to request H's information.
2. EMM retrieves the information from its memory and encodes the information using predefined strategies.
3. L generates the next input. The input is accepted. EMM varies the response time of the input to encode information. The EMM encodes a '0' to L if the response time $R_l \equiv T_l$, or a '1' to L if the response time $R_l \geq (i + 1) * T_l$, where i is an integer.

This scenario describes a timing covert channel. L deduces information from its varying response time $\{R\}$.

The aforementioned Exploitation Scenarios 1-5 require that an EMM has the confidential information L desires. The covert/overt channels launched by a malicious EMM can be prevented if an EMM contains no confidential information about H.

The following exploitation scenario assumes that the EMM is not malicious. H and L collaborate to transfer confidential information. We assume H's message is always processed before all of L's messages and an EMM will not halt if there are pending messages in its input or output buffers.

Exploitation Scenario 6:

1. L sends an input message to the EMM.
2. L waits for its response and then computes its response time R_l . It can calculate the number of H's messages processed by the EMM, assuming the processing time of each message is the same for both H and L. The number of H's messages is $N_h = R_l - T_l / T_h$.
3. There is an alternative way to calculate the number of H's messages. The number of H's messages processed by an EMM is the difference between all the messages an EMM computes and the number of responses, N_s , that L receives during a period of time t , $N_h = (t - (N_s * T_l)) / T_h$.

¹ A system is input total if it is always ready to accept inputs.

In a two-user system, L can clearly deduce this information from H. This channel can be reduced by adding noise to the system. If there is a third user in the system and L cannot distinguish their behaviors, the capability of the channel is reduced to an acceptable level.

3. Specifying EMM Security Requirements

Based on the aforementioned exploitation scenarios, we need to take both overt and covert information flow paths into consideration for specifying an EMM's security requirements. We therefore consider a non-interference based security property [5]. The design of the security requirement of an EMM must prevent these scenarios from happening.

In short, the security requirement of an EMM is that there is no undesired information flow within an EMM even in the presence of a malicious environment. From the above analysis, we summarize the situations in which the observation of L can be interfered with, both explicitly and implicitly, by the behaviors of H or EMM and from which information can be leaked.

- L's message content is modified with confidential information (write down).
- L receives messages that have no corresponding request (downgrading of H's message).
- L's input message loses its response (upgrading of L's message).
- L's input messages are rejected or accepted depending on H's behaviors (non input-total).
- L's response time changes depending on H's behaviors (intentionally).
- L's response time changes depending on H's behaviors (arbitrarily).

The security requirements of an EMM are as follows. These requirements prevent all possible exploitation scenarios, summarized in Table 2, from occurring.

- R1: All L's input messages are independent of all of H's input and output messages.
- R2: All L's output messages are independent of all of H's input and output messages.
- R3: The security levels of all messages must be tranquil (the security level of a message will not change after it is assigned).
- R4: Response time is fixed with respect to a given input.
- R5: An EMM must be input total.

We apply separability [7] to model the security requirements of an EMM. We call an EMM that satisfies separability a multiple single level secure EMM (MSLS-EMM). An MSLS-EMM can process information with different security levels while keeping the information separated.

scenarios requirements	S1	S2	S3	S4	S5	S6
R1			✓			
R2	✓	✓	✓	✓	✓	
R3		✓	✓			
R4						✓
R5				✓		

Figure 2. A Mapping from Threat Scenarios to Security Requirements

This model is stricter than the security requirements of an EMM since it prevents information flow from L to H. However, there are benefits of using a generic security model: 1) there is no need for model validation in that the correctness and rigorousness of the model has been thoroughly scrutinized, and 2) the well-studied features, e.g., compositionality, can be directly applied to the system security design. We analyze the model's appropriateness in Section 4.

4. The Appropriateness of the Security Model

The strictness of a security policy on one hand increases the security of a system and on the other hand reduces the system's functionalities and performance. Leverage between the system requirements and security requirements should be carefully maintained. Among the confidential information flow proposed in the literature, *separability* is the strongest security property, which rules out all information flow within a system and provides logical separation for uses/processes with different security levels [6, 7]. In addition, the security property expressed in the model is preserved when composed with itself, and it retains another component's security property when composed with a security property other than separability. This feature facilitates the secure composition of an MSLS-EMM with other EMMs. An MSLS-EMM can be plugged in any multiple-level system to enforce the confidential security policies of interest without compromising the security of the system.

In this section, we demonstrate the appropriateness of applying the security model in an MSLS-EMM. We consider two situations in which information flow from L to H is required.

Situation 1: MSLS-EMM security requires information exchange between L and H.

Situation 2: Environment security requires information flow from L to H.

In Situation 1, we demonstrate that separation of information with different security levels in an MSLS-EMM is not unduly strict for the MSLS-EMM to fulfill its functionalities. In Situation 2, we show that a separability-

secure MSLS-EMM can enforce security policies which require information flow from L to H.

There are three cases in Situation 1 that require information flow from L to H within an MSLS-EMM, which violates separability.

Case 1. A low input message must be sent out through a high output channel.

Case 2. A high input message must be sent out through a low output channel.

Case 3. A message is received and sent through channels with the same security level, but policy enforcement computation needs to refer to the history of an MSLS-EMM's state information classified at different security levels.

The first two cases correspond to upgrading and downgrading, respectively, which are excluded from the MSLS-EMMs we consider and are not our concerns here. The last case is slightly complicated because it involves the properties of the security policy an MSLS-EMM enforces. We discuss the third case together with the environment security later.

In Situation 2, the environment security, the security policy an MSLS-EMM enforces, requires information flow from low to high. Can an MSLS-EMM enforce such an environmental information flow?

Let us consider the Bell-LaPadula security model as an example. We consider two operations, $write(s, o)$ and $read(s, o)$, in the model, where s and o represent the subject and objects, respectively. The security levels in the model are defined by the set $\{TS, S, C, U\}$, in which $TS > S > C > U$. A security label L_1 dominates L_2 if $L_1 \geq L_2$. A write/read access is allowed if the security level of the object/subject dominates the security level of the subject/object. The security level of each operation (i.e., the message) is determined by the source of the information. We enumerate the security levels of the authorized actions in the system in Figure 3, in which subjects and objects are represented by their security levels.

$write(u, ts) = u$	$write(u, s) = u$	$write(u, c) = u$
$write(u, u) = u$	$write(c, ts) = c$	$write(c, s) = c$
$write(c, c) = c$	$write(s, ts) = s$	$write(s, s) = s$
$write(ts, ts) = ts$		
$read(ts, ts) = ts$	$read(ts, s) = s$	$read(ts, c) = c$
$read(ts, u) = u$	$read(s, s) = s$	$read(s, c) = c$
$read(s, u) = u$	$read(c, c) = c$	$read(c, u) = u$
$read(u, u) = u$		

Figure 3. Security Levels of Primitive Operations in BLP Model

Each message (or action) is sent to an MSLS-EMM through the input channel defined at its level. The MSLS-EMM enforces BLP by validating the domination relation of the security levels between the subject and object encoded in a message. Legal messages are sent out via an output channel with the same security level. An MSLS-EMM can securely enforce the policy.

Note that, for example, a TS user can receive information from all the output channels in which information flows from a lower level output channel to a higher level subject. This information flow, based on our definition, is environment information flow and therefore does not violate MSLS-EMM security. An observer cannot deduce any confidential information from the observation of the interfaces of an MSLS-EMM. The interaction between an MSLS-EMM and its environment is depicted in Figure 4. Outside the dashed lines is environment information flow while inside the dashed lines is MSLS-EMM information flow.

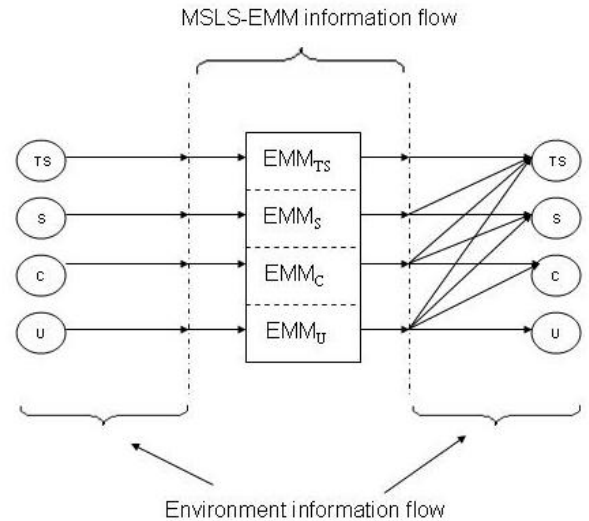


Figure 4. An MSLS-EMM enforces BLP.

Case 3 in Situation 1 requires information flow from L to H in the MSLS-EMM in order for it to enforce the environment security policy. This case can be further divided into two subcases: 1) policy enforcement computation needs to refer to the past actions of the untrusted programs classified at the same security level, and 2) policy enforcement computation needs to refer to the past actions of the untrusted programs classified at a different security level than that of the current message. Only the enforcement of security policies described in Subcase 2 demands information exchange between historical actions with different security levels. It is this type of security policies that is not enforceable by a separability-secure MSLS-EMM.

An MSLS-EMM can enforce all confidential security policies with no history reference and confidential security policies that require no inter-level history reference.

5. Conclusion

In this paper, we analyzed the security requirements of an EMM and view an EMM as part of the system it protects. A system is secure if both the system security policy is respected and the EMM faithfully enforces the policy. We applied an existing security model separability to model the security requirements of an EMM enforcing confidentiality in a multi-level system. An MSLS-EMM can enforce all confidential security policies with no history reference and confidential security policies that require no cross-level history reference.

We discovered that the functionalities of downgrading and upgrading must be considered exclusively from a normal EMM for the purpose of security. An MSLS-EMM must maintain the tranquility of the security level of messages it has sanitized.

An MSLS-EMM will preserve the system security when it is deployed in a target system and EMM security when it is composed with other MSLS-EMMs, which are guaranteed by the compositionality of separability [7, 13]

This work can be used for enforcing confidentiality in high assurance embedded systems [1] extensible systems, etc. Recent work in language-based security mechanisms [8] indicates that we could securely implement an MSLS-EMM.

Acknowledgment

This material is based on research sponsored by AFRL and DARPA under agreement number F30602-02-1-0178. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expresses or implied, of AFRL and DARPA or the U.S. Government.

References

[1] J. Alves-Foss, W. S. Harrison, C. Taylor, and P. Oman. The MILS architecture for high-assurance embedded systems. *International Journal of Embedded Systems*, in press, 2006.

[2] J. Alves-Foss, C. Taylor, and P. Oman. A multi-layered approach to security in high assurance systems. *Proceedings of Hawaii International Conference on System Science*, Hawaii, Jan 2004.

[3] L. Bauer, J. Ligatti, and D. Walker. More enforceable security policies. *Proceedings of the IEEE of Computer Security Foundations Workshop*, July 2002.

[4] P. W. L. Fong. Access control by tracking shallow execution history. *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, May 2004, 43-55.

[5] J. A. Goguen and J. Meseguer. Inference control and unwinding. *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, May 1984, 75-86.

[6] H. Mantel. Possibilistic definitions of security - an assembly kit. *Proceedings of the IEEE Computer Security Foundations Workshop*, July 2000, 185-199.

[7] J. McLean. A general theory of composition for trace sets closed under selective interleaving functions. *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, May 1994, 79-93.

[8] A. Sabelfeld and A. Myers. Language-based information-flow security. *IEEE Journal on Selected Areas in Communications*, 21(1), 2003.

[9] F. B. Schneider. Enforceable security policies. *ACM Transactions on Information and Systems Security*, 3(1), Feb 2000, 30-50.

[10] J. Son and J. Alves-Foss. Covert timing channel analysis of rate monotonic real-time scheduling algorithm in {MLS} systems. *Proceedings of the IEEE Information Assurance Workshop*, West Point, NY, June 2006, 361-369.

[11] M. Viswanathan. *Foundations for the Run-time Analysis of Software Systems*. PhD thesis, University of Pennsylvania, Dec 2000.

[12] B. Wang. *Possibilistic information flow analysis and formal verification of multiple single level secure execution monitoring mechanisms for high assurance systems*. Master's thesis, University of Idaho, May 2006.

[13] A. Zakinthinos and E. Lee. A general theory of security properties. *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, May 1997, 94-102.

[14] S. Zdancewic. A type system for robust declassification. *Proceedings of the Conference on the Mathematical Foundations of Programming Semantics*, Electronic Notes in Theoretical Computer Science, Mar 2003.