Secure Splitting Block (SSB)

Libor Kratochvíl

libor.kratochvil@i.cz

S.ICZ a.s.
V Olšinách 75
100 97 Prague, Czech Republic

Abstract

Data created by today's information systems are not sufficient for their proper operation. Direct access of such systems to public data is needed. Systems often provide services to external subjects on-line as well. These features include communication with an external environment, which is usually considered insecure within the given system's security policy. Developers have to solve the problem of how to assure communication without connecting the system to an insecure environment. The following paper describes a possible technical solution to this problem based on a Secure Splitting Block device, an alternative to the commonly used firewall technology.

The SSB is device designed to exchange data files between isolated networks. Files are transferred without the need for physical transport of external data media while secure separation of communicating peers is preserved. The SSB may be applied in separated environments, especially those processing secret information. SSB's main benefit is the possibility to run distributed applications requiring frequent data updates in a separated environment. Thanks to its architecture, implementation of the device into existing systems does not require a modification of their security policy, as long as physical media exchange was used previously. Built-in audit mechanisms also lower the risks associated with human factor within physical media exchange.

Keywords: communication, interconnection.

1 Introduction

Building a contemporary Information System is always a compromise between functionality and security of a given solution. System architects oscillate between the two extremes of building a 100% secure system providing very little information and building a 100% user-friendly, insecure and open system. The functionality value of both of the above extreme options would be approximately the same - zero. Connection of an insecure system to a public environment (especially the Internet) would mean immediate attacks - especially the simplest DoS-type attacks. The owner of such a system would also have to face the legal consequences of leak and misuse of system information.

Another problem is the fact that only a very small part of information systems is capable of operating solely with data created by the system itself, or data entered into the system during commissioning. This of course also applies to systems processing confidential information, where the situation with input and output of information to/from the system is even more complicated because it is subject to measures authorized by the relevant national security agency.

Generally speaking, two basic types of solutions are used for secure data communication between information systems today. These two solutions are at the opposite sides of the spectre for both functionality and security. The first approach involves a controlled interconnection of systems directly at the network level. Interconnection control is based on filtering of communication protocols and/or implementation of application proxy gates. Configuration of the filtering element provides the required level of functionality and security across the entire spectre - from full interconnection to complete prevention of traffic. Such approach offers very low guarantee of preventing unwanted interconnections. Such connections may result from several causes, for example configuration errors, communication protocol errors or as a result of exploiting a interconnection device implementation error, because the transferred data interact directly with the interconnection device system. Such errors may result in a break of system's security and/or leak of secret information. For these reasons the only application of this type of interconnection within secret information processing systems is in systems processing secret information at the same security level, see for example [1].

The second type of approach to communication between the system and the environment is to physically divide the system from the outside world and exchange information manually using physical transport of external data media. This approach provides a high level of security, but minimizes functionality. Its advantage is the fact that errors in configuration, interconnection device, or communication protocol are ruled out, but it introduces the human factor into the information exchange, which influences the channel error rate. The use of humans also limits audit functionality. The biggest limitation of this type of 'communication' is the fact that its latency does not allow for use with distributed applications dependent on frequent data exchange. This type of transfer is currently being used for limited data exchange between systems with different security levels.

The purpose of this paper is to introduce a system architecture of a device which would be capable of combining the advantages of both the above approaches to secure communication. The paper proposes requirements and evaluates the possibility of implementing a device enabling continuous, low-latency transfer of large volume of data without interconnecting the communicating parties. Such device could solve the problem of automated data exchange between systems working at different security levels, or systems maintained by separate and distrustful authorities.

2 Requirements

The Introduction above contains the basic concept of a device enabling data communication, which meets the following contradictory requirements:

- Secure separation of communicating parties.
- Frequent exchange of large data volumes.

These requirements, seemingly impossible to fulfil at the same time, may be, under certain circumstances, implemented into a device enabling secure data exchange while reliably separating the communicating entities. The architecture concept must be based on the premise of developing a *splitting*, rather than an *interconnection* device. Implementation of such a device *MUST NOT* interconnect the two communication systems at any of the OSI model layers. The design must primarily concentrate on all the splitting functions, only secondarily on any transmission functions. For the purposes of this paper, the proposed device shall be referred to as Secure Splitting Block (SSB).

2.1 Communicating Entities

As for any design, the area in which the solution will be used must be defined first. For the SSB, the area of use is two general subjects exchanging data but not wishing to be interconnected. No security requirements for these two subjects exist as far as the SSB development is concerned. They could be any two independent information systems. The design aims to provide such a level of guaranteed security to make SSB suitable for bi-directional data exchange even between systems operating at different security levels.

2.2 Design Concept

The design concept was already defined by the above article. Any device to be connected to a system with a defined security policy must be entirely controlled by the authority responsible for that system. This requirement determines that the SSB device must consist of at least two independent but completely equivalent parts (blocks). Each of these parts is controlled by one authority, providing all control functions, including physical access, to that authority only.

An accurately defined interface must be provided between these two parts, providing only a pre-defined minimum and limited set of functions necessary to ensure the required functionality. The interface must be designed and implemented so that it respects complete independency of both connected SSB blocks and so that monitoring or controlling the operation of the opposite block or even the system to which it is connected is not possible. Such interface provides the “Point of Segregation“ of both systems.

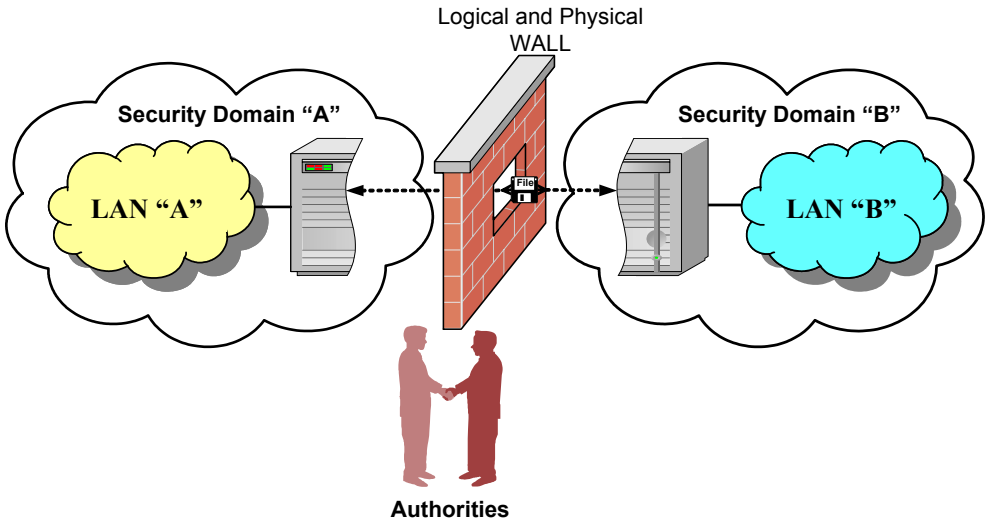

Figure 1: SSB Philosophy.

The device must provide a functionality equivalent to the following: data to be transferred are first saved to a pre-determined computer within the network and then copied to a data medium as a file. The medium is then handed over to the opposite communication party, where the data is re-introduced into the system using a reversed order of action. From the above definition it is clear that the communicating parties will not be interconnected by any communication protocol and **all data will be transferred in an unstructured data file which does not interact with the SSB system itself**.

2.3 Design Principles

As for any design process, a set of basic rules, which the final product must meet, should be defined prior to commencement. For the implementation of SSB, these are at least the following:

1. **The Technology Diversity Rule** - To provide a high level of guaranteed security, the possibility of any arbitrary error causing a failure of the implemented security mechanisms resulting in the connected system being penetrated must be completely ruled out. The device must therefore consist of a minimum of two different types of hardware and software.
2. **The Distrust Rule** - The solution must not - in any of its parts - be dependent on the correct function of a sole security mechanism. A possibility of failure must be expected for all implemented parts - especially as far as its influence on failures of the secure splitting functions is concerned. A failure of a transfer function and subsequent loss of availability is not concerned a risk in this context. On the contrary, it is desirable that the eventual failure of any of the splitting functions results in a shutdown (inoperability) of the entire SSB device (instead of a possible penetration of the connected subject)**.**
3. **The Multiple Mechanism Rule** - The implemented security mechanisms must be multiplied to the greatest possible extent, on the principle of (technically) different applications of the same mechanism in different parts of the device. A cascade of technically identical mechanisms with the same functionality is not considered a multiple mechanism.

The above "Technology Diversity" rule already pre-defines the block diagram of the entire device. The device must consist of two identical parts (see "Design Concept") and each of those parts must contain a minimum of two different hardware components. See "SSB Block diagram" on the figure below.

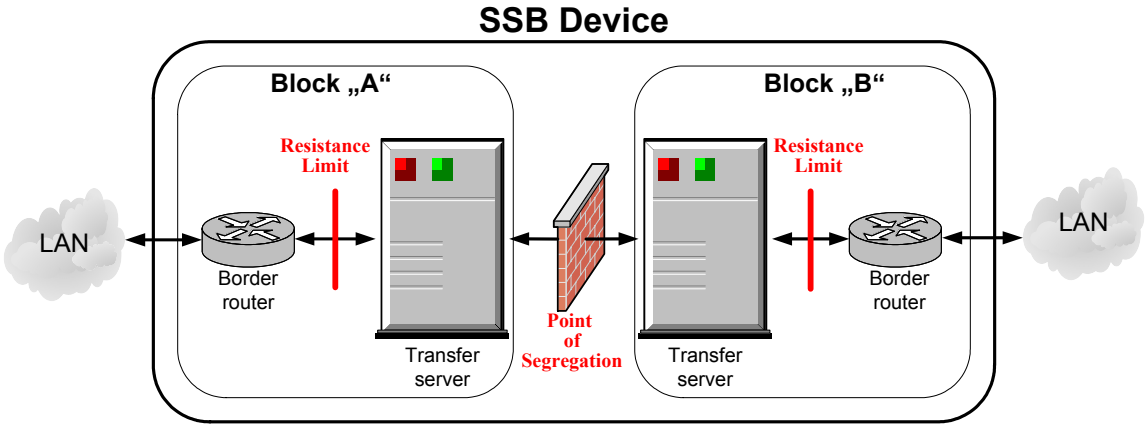

Figure 2: SSB Block Diagram.

Assessment of “Resistance Limit“ is based on the “Distrust Rule“ maximized to contain possible complete breaking of security of one of the used technology platforms. The “Resistance Limit” defines the maximum possible extent of penetration of the entire SSB device, which does not result in penetration into the network connected to the SSB.

The “Multiple Mechanism Rule“ and the fact that the solution is to be used for communication, defines the main system components. Since this is a communication device, one of its technology elements is a router, which provides both classic router functionality and usual firewall functions. The second platform must provide the required functionality and at the same time offer safety mechanisms similar to those of a normal firewall. A UNIX-type operating system server fulfils these requirements.

2.4 Data Transfers

The process of data transfers from one system to the other must consist of two completely separate activities. The first is communication between a networked computer and the SSB device itself. This communication uses standard communication protocols and its goal is to transfer the file between the network client and the SSB device block attached to it. This communication must comply with several requirements:

- All transfers may only be realized from authorized network nodes.
- Only authorized users may initiate data transfers.
- The communication must be cryptographically protected.
- The transfer protocol used must be a status-type protocol.
- All data transfers must only be initiated by a networked computer, never by the SSB device itself.

The second part of the transport process consists of the data file transfer between the SSB device blocks. This data transfer must be completely independent on processes within the two information systems connected to the SSB and on communication of the networked stations with the SSB device. The connection between both SSB device blocks must not use any network protocol or be controlled by any privileged process. The connection may be implemented using a simple serial link, controlled by a non-privileged process with the transferred files on a data medium as the only output and input. The following requirements must be met:

- The communication is completely independent and asynchronous.
- The transfer link does not allow any other services apart from single bytes data transfer.
- The data flow control is provided at hardware level.
- The transfer interface (“Point of Segregation“) is implemented in the controlling process.

2.4.1 Transfer Queue Architecture

For maximum functionality, the device must enable full duplex data transfers. Because the processes transferring data between the SSB device blocks must not be influenced by the clients-to-SSB communication, receipt and sending of data between SSB and network clients must be logically separated.

We may also expect the device to be used by more than one subject within each of the information systems attached to it. It will therefore operate several logical data flows. SSB must therefore include functions enabling division of individual data flows, their individual settings and control of user access. A communication queue model may be used to meet these requirements as shown on the figure below.

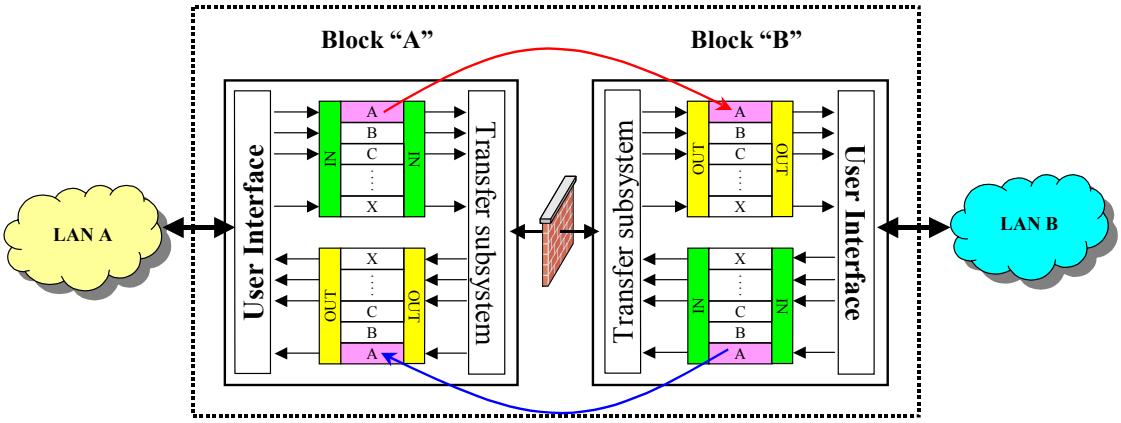

Figure 3: SSB Transfer Queues.

FIFO-type queues are most suitable for SSB purposes. Two queues for one duplex data flow at each SSB device block will be used. The queues form a transfer channel between the processes providing communication with clients within the networks attached to the SSB and processes providing transfer of data to the opposite block. To keep the analogy between the SSB and the data media transfer method as close as possible, the queues must be implemented by using a data medium. This provides another division in the communication path and prevents any direct network connection between both communication parties. The following conditions must be fulfilled during practical implementation:

- Only regular data files are saved into the queues as they were received from the user. No meta information on their contents or subsequent processing is created or transferred.
- Processes at opposite ends of the queues do not share any configuration files.
- Processes at opposite ends of the queues do not communicate with each other - synchronization must only be based on the principle of a presence of a transferred file in the queue.

2.4.2 The Transfer Link and "Point of Segregation"

The serial transfer link must use a new type of connection hardware, which was not used as a network interface by the operating system before. The risk of a "forgotten code", which could be used to implement a network protocol at this communication link, must be eliminated.

The "Point of Segregation" must be defined and implemented so that it expects possible attacks from the opposite SSB block administrator. All communication across this point must commence and be completed as a data file on a computer hard disc. This eliminates any eventual attacks to the level of the regularly transferred files with which the system does not interact further in any way. Because of the "Distrust Rule", possible corruption of the process controlling the transfer line must be expected and relevant measures taken during development. The "Point of Segregation" must only provide the following functions:

- Sending of the request for file transfer and data file sending.
- Receipt of the request for file transfer and data file receiving.

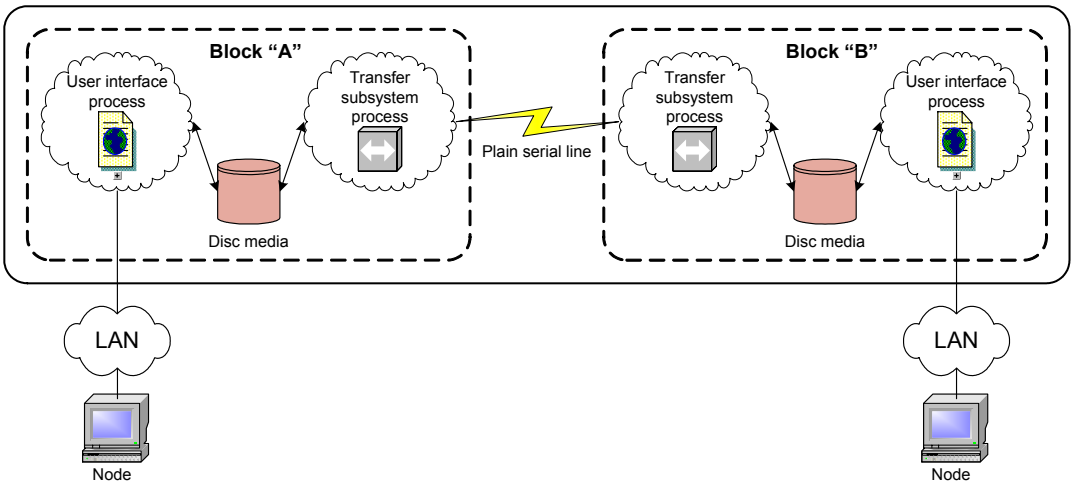

Figure 4: SSB Data Path Implementation.

2.5 Device Administration

The device architecture itself determines that its administration must be done by at least two independent authorities. When taking the "Distrust Rule" into account, it is also evident that device administration must be delegated to several administrators within each of the responsible authorities. The block diagram clearly shows a division between the administration of the border router and the transfer server. The transfer server consists of the operating system itself and of application processes providing the required functionality. System administration and application administration should therefore also be divided. Three types of administrators will therefore be necessary:

- Border router administrator
- Transfer server operating system administrator
- User interface application administrator

Because the SSB is a device providing a high level of security, special requirements are placed on administration. First, remote administration of the SSB device must not be possible. The user interface application administration is an exception to this rule, because user logon is subject to conditions other than account settings in the application only (cipher-key, filter settings). All application processes also run in a secure environment under a non-privileged account. Any eventual configuration error or application administrator account misuse is therefore not capable of influencing the operation of other device parts, or penetrating the network attached to the SSB.

The SSB design concept requires it to function as an analogy to the process of data transfers using external data media, whose principle itself guarantees that both communication parties will not be interconnected in any way. The system will thus always provide a certain minimum security level. This puts special requirements on the system administration. System administration generally assumes the existence of a privileged user with no limitations from the system. This principle is not acceptable for a SSB-type device. During normal SSB operation, the administrator must not have unlimited user rights. The administrator is only allowed to implement actions that cannot influence system security, which in reality means monitoring from the system console only.

The administration tasks themselves may only be implemented when the system is in a special operation mode, under which no application or transfer subsystems are running and may not be explicitly started either. The device administration must be automated to a maximum level and must consist of a minimum number of pre-defined steps to prevent possible errors. For emergency situations, which are not defined under the normal administrator's tasks, all available tools may be used to repair the system, but the probability of an error increases. Transition to normal operation mode must be ensured after any administration. This may only be done by restarting the transfer server's operating system. Such restart must ensure that the system will enter a

consistent state after resuming operation, that is eventual administration errors will not have influence on normal system operation.

2.6 Audit and Archival

A SSB-type device must ensure a reliable audit of all actions and as a bonus provides archival of transferred data. The archival should be implemented as an optional parameter of the transfer queue configuration, because it may not be necessary to archive certain data flows during normal operation. This will save space on the archiving medium and therefore prolong the operating period between maintenance tasks. The data archival will have significant requirements on the data archival space and will be the most frequent cause of administration tasks.

Operation of the device must be conditioned by correct functioning of the audit subsystem. A situation when the audit subsystem is inoperative and the device still transfers data is not acceptable. The audit functionality must not be configurable and all subsystems must verify correct audit functionality prior to any activity. The following special requirements must be put on the audit and archival during implementation:

- Data are saved to an external medium which may be taken out of the device at any time.
- Data writing must only use "append-only" mode, if possible at the level of physical writing to the medium (equivalent to CD-R).
- Integrity of audit data on the medium must be verified during system start-up.

3 Solution Proposal

Definition of all requirements is only the first step in a successful solution of a problem. To ensure the required assurance level, all requirements defined above must be correctly implemented using existing technology. If possible, some requirements should be extended. Several rules must be defined prior to development itself. These rules will significantly influence the resulting solution:

1. **The Transparency Rule** - concerns the transfer server and defines that none of its parts may contain a "black box", everything must be verifiable in detail (does not concern hardware components). In reality this means that all software must be available in source code.
2. **The Minimizing Rule** - defines that all design proposals must minimize functionality of individual components so that only the minimum required overall functionality is provided.
3. **The Assured Integrity Rule** - defines that the system must always be at a defined minimum integrity level, which may not be corrupted. Prior to operating the system, full integrity must be ensured.
4. **The Non-compatibility Rule** - defines the tendency to implement technologies with the highest possible non-compatibility level with current standards.
5. **The Rule of Limits** - defines that the system acts as a finite automaton in all of its parts. It is possible to simulate and test all possible conditions the system may be in. The same applies to all configurable parameters.

3.1 Border Router

Besides router-functionality, it must also provide reliable IP filtering. The border router's goal is to:

- Limit access to the SSB to pre-defined IP addresses.
- Protect the transfer server from attacks from the connected network.
- Protect the connected network during eventual corruption of the transfer server system by implementing the "Resistance Level"

The router must be configured according to the "Minimizing Rule" and must meet the following conditions:

- Remote administration or monitoring of the system is not allowed.

- Filters allow connections to be initialised in one direction only - from the connected network to the SSB.
- Packets may only be transferred to the SSB's user interface application.
- Filters must be provided at all router interfaces and defined for both communication directions.

3.2 Transfer Server

The transfer server's operating system was defined by the "Transparency Rule", which is applicable only to free-distributed versions of UNIX – OS's, such as LINUX. Of course we can not just use one of the standard distributions, extend it with the required functionality and put it into the system. For the above rules to be fulfilled and the required assurance level ensured, implementation of the operating system must meet the following conditions:

- The entire system must be located on and run from a "read-only" medium (CD ROM) – "The Assured Integrity Rule"
- The system must contain necessary components only – "The Minimizing Rule"
- Source code must be available for all binary modules and the operating system's kernel itself – "The Transparency Rule"
- Applications can not be run and special devices used from write-enabled media – "The Assured Integrity Rule"
- The maximum continuous system operation period must be limited – "The Rule of Limits"
- An administrator logged in at the server console is the only interactive user.

3.2.1 Operating System Kernel

Compared to normal OS's, the kernel of the SSB's operating system must also meet special requirements. The main difference in the SSB functionality is the requirement to restrict some OS kernel functions to all users, including the administrator. The kernel must therefore contain some type of a security subsystem which enables definition of restrictions whose application is otherwise not possible in an operating system. Implementation of this subsystem must ensure that:

- The security subsystem can not be switched off during normal operation of the device.
- For administrative reasons, it must be possible to switch the operating system into a "security subsystem disabled mode", but start of pre-defined processes must be disabled in this OS mode.

The OS core must be built to suit the "Minimizing Rule" and fitted for the concrete hardware. The kernel must consist of a single compact module (static-linked), which may not be extended to include other functionality during system operation.

3.2.2 Security Subsystem

Besides restricting the administrators and providing extra security configuration possibility for the operating system itself, the security subsystem must also enable implementation of "The Multiple Mechanism Rule". This rule also defines that the SSB operating system must not be left unprotected during an eventual failure of the security subsystem (see the "Distrust Rule"). Its required functionality may be summed up as follows:

- Administrator restriction.
- System calls restriction.
- Restriction on identity change for critical processes.
- Higher granularity of the access control to the file system.
- The "append-only" function for log files.
- Restriction of access to network services.

- Restriction of start of applications from write-enabled media.

3.2.3 SSB Application Processes

Implementation of application processes will influence the final level of separation offered by the SSB. The separation is already provided by inclusion of a plain serial link and a data medium into the transfer path. Incorrect implementation of service processes may create hidden alternative connections outside of the data medium (with is supposed to be the only connection between those processes) in each block. In keeping with the "Multiple Mechanism Rule" and the "Technology Diversity Rule", the data medium is not the only element ensuring secure separation. Eventual failure or a design error will therefore not result in interconnection of the communicating systems. Secure separation would then still be provided by border routers and especially the serial link providing the "Point of Segregation".

Application processes must be highly resistant to their possible use for penetration of remaining system parts. Generally and according to the "Distrust Rule", the possibility of penetration of individual running processes is taken in account, but the level of such penetration relative to the entire system must be limited to an isolated environment in which the relevant process is running. In reality this means that individual application processes must be run in divided environments with a minimum amount of available assets. Such environments must be strictly divided from each other and no application process may be capable of concurrent or gradual access to more than one of them. The "chroot"-mechanism can be used in practical implementation of such environments and the security subsystem will add further functions to this mechanism.

Implementation of the "Technology Diversity Rule" also requires that the binary modules of the individual process groups are implemented using different techniques. That means part of the processes will be linked statically and part of them dynamically and different libraries and compliers will be used in their implementation. The following rule was defined to provide extra separation security:

1. The **Simplex Transfer Rule -** is defined for data exchange between the SSB application processes and requires that data file transfers between application processes at each of the SSB device blocks may **only be one-way.**

The above requirement on data transfers shows that application processes at each of the SSB device blocks must be divided to a minimum of two absolutely independent groups which must not influence each other at all. However, the "Simplex Transfer Rule" requires a minimum of three communicating subjects.

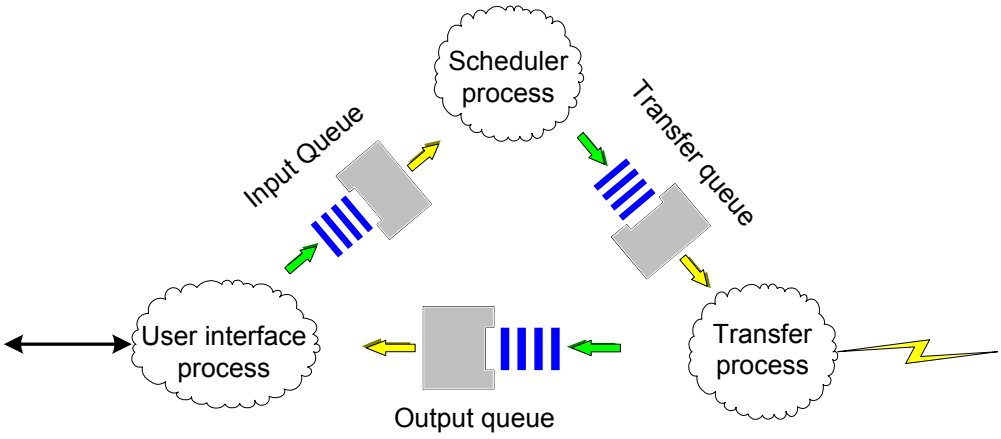

Figure 5: SSB Interprocess File Flow.

3.2.3.1 User Interface Processes

This process group ensures communication between the SSB and the users of the connected network and provides the possibility of transfer of data files between the users of the connected network and the queues at the SSB. The processes must ensure user access control for this application and to individual transfer queues, authentication of network stations and encryption of the transfer channel between the SSB and the

communicating client. Practical implementation of all of these requirements should use a Web server-based solution with SSL communication support. This solution fulfils all the rules defined in this paper.

The most important security factor of this group of processes as far as secure separation is concerned is the prevention of their access to the serial communication line connected to the opposite block.

3.2.3.2 Transfer Subsystem Processes

This process group provides secure transfer of data files from the hard disc to the opposite SSB block. These processes have two main goals. The first is the secure implementation of the "Point of Segregation", to ensure that the serial line may not be used to access the system's network services or to corrupt them. The second goal is to plan individual data transfers depending on the configuration of transfer queues. Division of these processes into two separate parts ensures compliance with the "Simplex Transfer Rule" and increases the secure separation assurance level.

The most important security factor of this group of processes as far as secure separation is concerned is the prevention of their access to the system's network services.

4 Implementation Example

The example demonstrates the possibility of data exchange between information systems with different security levels. A typical example from intelligence services environment is the need to transfer data between an agent in the field and a secret information system. Such agents usually transfer encrypted messages over an insecure channel into the intelligence services' central control room, where they are manually copied into the information system in which directions to be sent back to the agent are processed. Making the process automated using the current technology would involve a connection of the secret information system to insecure environment, which is not possible. Implementation of an SSB device based on the principles formulated in this paper enables the implementation of automated data transfer between the secret system and insecure environment and at the same time guarantees their secure separation.

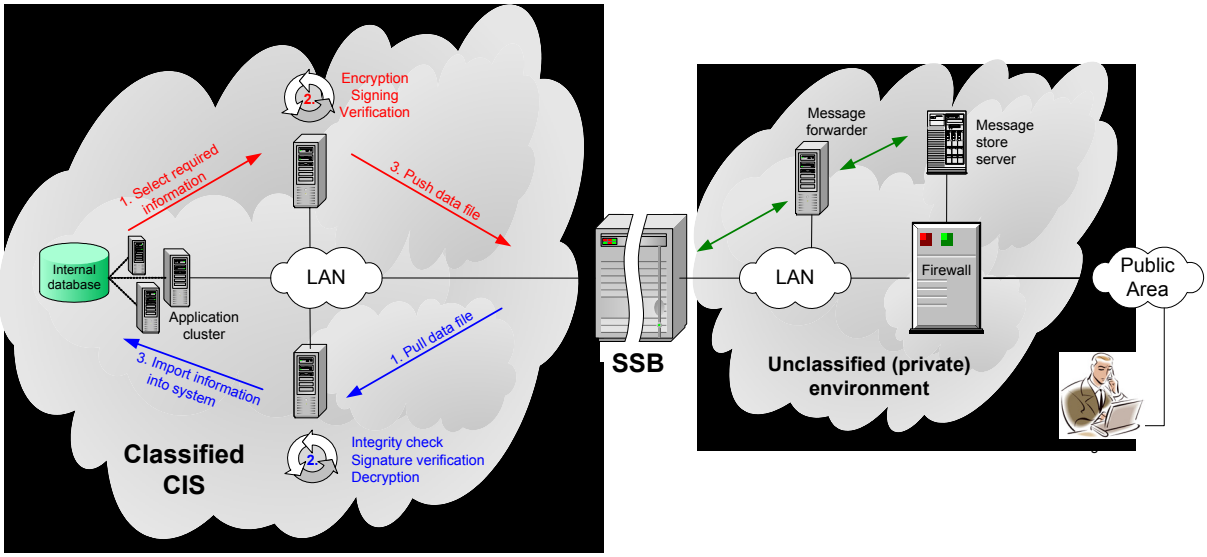

Figure 6: SSB Implementation Example.

The biggest problem arising from communication between systems operating at different security levels is data transfer from a system with higher security level to that with lower security level. Each piece of information transferred in this way must undergo a pre-defined process of lowering the security level at least to that of the target system. This process must be defined as part of the security policy of each system and implemented before the information is sent to the SSB. In our example, such process involves encrypting the directions returned to the agent using a certified method.

5 Conclusion

When a SSB-type device is successfully implemented, a new opportunity will open for communication between independent systems whose functionality was previously limited for security reasons. This paper shows that a suitable combination of current technology methods may provide a device capable of combining the application advantages of firewalls with the security of data transfers implemented using physical transport of external data media.

Of course such a device can not provide an universal solution to all problems, because secret systems will always be very specific. Those who expected that using a SSB-type device would enable them to access the Internet from their computers within secret information systems will probably be disappointed. But less demanding users will be able to use public data sources (for example press agency servers) and obtain large amounts of up to date information from them directly into the secret system. The biggest advantage of the solution presented here is the possibility to use distributed applications dependent on frequent data exchange across divided environments.

An important aspect that was not mentioned previously are the operational conditions of such a device. It is not possible to overcome all problems which prevented secure communication before just by connecting a SSB device into the system. The SSB solution should be understood as a combination of the device itself and its implementation into the system. Availability of the proposed device is therefore a necessary but not the only prerequisite of possible automated data exchange between systems operating at different security levels.

When contemplating practical use of SSB devices, one must also consider the fact that there will be no network connection between the communicating systems while most current distributed applications require such a connection. Each distributed application operating across the SSB will probably have to be altered to be able to work in a mode similar to that used in distributed applications where manual data transfer on external media is used.

The design concept requires that the SSB device does not interact with the transferred data in any way. From this requirement it follows that the SSB does not provide the functions of integrity and identification of origin of the transferred data. It also does not provide the function of confirmation of receipt of data (their collection by the user from the output queue). Implementation of such functions is up to the communicating subjects, because SSB is a device intended solely for the purpose of secure data transfer while maintaining a secure division between both communicating parties.

References

[1] INFOSEC Technical and Implementation Directive for the Interconnection of Communications and Information Systems, AC/322-D/0030-REV2, 25 October 2002 (NU).