ITT INDUSTRIES
DRAGONFLY GUARD
FINAL EVALUATION REPORT
VERSION 1.1

Victoria A. Ashby
Santosh Chokhani
James C. Reynolds

October 29, 1998

Suite 100 West ♦ 7927 Jones Branch Drive ♦ McLean, VA 22102-3305 ♦ 703 848-0883 ♦ Fax 703 848-0960

TABLE OF CONTENTS

TABLE OF CONTENTS ..... **II**

TABLE OF FIGURES ..... **IV**

TABLE OF TABLES ..... **V**

1 EXECUTIVE SUMMARY **1**

2 IDENTIFICATION **3**

3 SECURITY POLICY **4**

3.1 IDENTIFICATION AND AUTHENTICATION POLICY 4
3.2 MANDATORY ACCESS CONTROL POLICY 4
3.3 DISCRETIONARY ACCESS CONTROL POLICY 5
3.4 AUDIT POLICY 5

4 ASSUMPTIONS AND CLARIFICATION OF SCOPE **8**

4.1 USAGE ASSUMPTIONS 8
4.2 ENVIRONMENTAL ASSUMPTIONS 8
4.3 CLARIFICATION OF SCOPE 9
4.3.1 Single Dragonfly Guard Between Two Domains *9*
4.3.2 A Dragonfly Guard for each Domain *10*
4.3.3 A Complex Configuration *11*
4.3.4 Tunneling between Low Domains Through a High Network *14*
4.3.5 Using Dragonfly Guards with Firewalls *17*
4.3.6 Military Network Configuration Issues *17*

5 ARCHITECTURE **18**

5.1 SYSTEM OVERVIEW 18
5.2 HARDWARE OVERVIEW 18
5.2.1 CPU Board *18*
5.2.2 Flash Floppy *19*
5.2.3 Ethernet Interface *19*
5.2.4 PCMCIA Reader *19*
5.2.5 Fortezza Card *19*
5.2.6 Ignition Card *19*
5.2.7 RS-232 Port *19*
5.2.8 Power Supply *19*
5.2.9 Case *20*
5.3 SOFTWARE OVERVIEW 20
5.3.1 Boot and Self Test *20*
5.3.2 Software Load and Validation *21*
5.3.3 Dragonfly Initialization *21*
5.3.4 Dragonfly Operation *22*

6 DOCUMENTATION **23**

6.1 DRAGONFLY GUARD USER MANUAL 23
6.2 INSTALLATION CARDS 24

7 PRODUCT TESTING **25**

7.1 ANALYSIS OF VENDOR'S TESTING EFFORT 25
7.1.1 Details of Test Suite 25
7.1.2 Test Configuration 26
7.1.3 Coverage and Depth Analysis 27
7.1.4 Testing Approach 27
7.1.5 Results of Vendor Testing 27
7.2 EVALUATION TESTING 28
7.2.1 Test Configurations 28
7.2.2 Rerunning Vendor Tests 28
7.2.3 Independent Tests 29

8 EVALUATED CONFIGURATION 31

8.1 EVALUATED HARDWARE AND SOFTWARE COMPONENTS 31
8.1.1 Evaluated Hardware Components 31
8.1.2 Evaluated Software Components 32
8.2 CONFIGURATION AND USAGE NOTES 33
8.2.1 Required and Allowed Configuration Settings 33
8.2.2 Non-Evaluated Configuration Settings 34
8.2.3 Incorrect Installation of the Evaluated Configuration 34
8.3 TARGET ENVIRONMENT 35
8.4 RESIDUAL VULNERABILITIES 35

9 RESULTS OF EVALUATION 36

9.1 TOE SECURITY FUNCTIONAL REQUIREMENTS 36
9.2 STRENGTH OF FUNCTION (SOF) REQUIREMENT 49
9.3 TOE SECURITY ASSURANCE REQUIREMENTS 50

10 EVALUATOR COMMENTS/RECOMMENDATIONS 61

11 ANNEXES 62

ANNEX A: DRAGONFLY GUARD ADMINISTRATION USER MANUAL 62

12 SECURITY TARGET 63

13 GLOSSARY 64

14 BIBLIOGRAPHY 65

14.1 DRAGONFLY DOCUMENTS 65
14.2 GOVERNMENT DOCUMENTS 65

TABLE OF FIGURES

FIGURE 4-1: SINGLE DRAGONFLY GUARD BETWEEN TWO DOMAINS 10
FIGURE 4-2: DRAGONFLY GUARD FOR EACH DOMAIN 11
FIGURE 4-3: A COMPLEX CONFIGURATION 13
FIGURE 4-3A: TUNNELING THROUGH A SECRET NETWORK 14
FIGURE 4-3B ANOTHER MIXED ENCLAVE EXAMPLE 15
FIGURE 4-4: USING THE DRAGONFLY GUARD WITH A SEPARATE FIREWALL 16
FIGURE 5-1: DRAGONFLY HARDWARE ARCHITECTURE 20
FIGURE 7-1: VENDOR'S TEST CONFIGURATION 26

TABLE OF TABLES

TABLE 3-1. ANTICIPATED MESSAGES ..... **ERROR! BOOKMARK NOT DEFINED.**
TABLE 4-1. SECURE USAGE ASSUMPTIONS 10
TABLE 8-1. REQUIRED CONFIGURATION OPTIONS 38
TABLE 8-2. OPTIONAL CONFIGURATION SETTINGS 39
TABLE 8-3. NON-EVALUATED CONFIGURATION SETTINGS 39
TABLE 9-1 EAL2 ASSURANCE COMPONENTS 66

1 Executive Summary

The Dragonfly Guard Model G1.2 is a network security device produced by ITT Industries. A Dragonfly Guard is a simple rugged box, roughly the size of an external modem, containing a 486 motherboard. The unit has two Ethernet interfaces, a serial port, and two PCMCIA card slots. It requires two cards to operate. The first card is the Ignition Card that contains digitally signed Dragonfly software release 3.0. The second card is a Fortezza Card with several digitally signed certificates containing network configuration information.

Dragonfly Guards use National Security Agency (NSA) Fortezza Cards to provide multi-level secure (MLS) services to Internet Protocol (IP) networks. The Dragonfly Guard operates on standard IP datagrams. The Dragonfly Guard provides the following security services: mandatory access control, discretionary access control, confidentiality, integrity, source authentication, and audit. The Dragonfly Guard cryptographically labels every IP datagram with an appropriate security level, and then checks that label before releasing the underlying datagram in plaintext form. The Dragonfly Guard provides discretionary access control between the Domains that it protects. All User Data is encrypted and integrity checks are applied to all messages transmitted between two Dragonfly Guards. The Dragonfly Guard can also serve as a firewall or an in-line encryptor. In order to provide these services, Dragonfly Guards set up a trusted Association based on source authentication and use the Fortezza Key Exchange Algorithm to generate a symmetric key. Any Dragonfly Guard can also be designated as an Audit Catcher. Audit Catchers receive audit reports from other Dragonfly Guards and send all messages to their serial port for printing, storage, or subsequent analysis. The selection of auditable events can be set by an Audit Mask.

Dragonfly Guards separate two Dragonfly Domains. A Dragonfly Domain is a set of computers that are networked together without any intervening Dragonfly Guards. These computers in the same Domain may be PCs, Workstations, or Servers that are all at the same security level.

Dragonfly Guards always authenticate themselves to each other. All Dragonfly Messages sent before an association is formed or outside of an Association are digitally signed. This includes Association Requests and Association Grants. After an association if formed, messages are encrypted with a symmetric key known only to the source and destination Dragonfly Guard.

The Dragonfly Guard supports Mandatory Access Control (MAC) by labeling every IP Datagram with an appropriate security level. It then checks that label against the security level of the destination Domain before releasing the underlying datagram in plaintext form to the destination host. Through the sharing of security related information via an Association, Dragonfly Guards can support both Write Equal and Write Up. In the Write Equal environment, where Dragonfly Domains are at the same security level, all IP based communications are allowed according to the MAC policy. Dragonfly also allows transfer of User Data from a low-level Domain to a high level Domain called Write Up.

In the case of Write Up, Dragonfly supports only the subset of IP based functionality for which the Dragonfly Guard can predict the response. Many IP-based protocols require some form of feedback.

For example, the file transfer protocol (FTP) uses flow control. The feedback constitutes a potential Write Down. Dragonfly assures that this Write Down does not constitute a violation of the security policy by a patented scheme of anticipated messages. Each feedback message is predicted by the Dragonfly Guard based upon the Internet Control Message Protocol (ICMP) or Domain Name System (DNS) request, or the allowed Write Up FTP or Simple Mail Transfer Protocol (SMTP) command. If the actual message matches the predicted message, except for certain fixed length control fields such as sequence number and window size, the predicted message is released with the control field data from the actual message copied to the predicted message. Otherwise, no message is released and there is no feedback.

The Dragonfly Guard uses Privilege Vectors for Discretionary Access Control (DAC) between Domains. All communication allowed by DAC is bi-directional. Therefore, if the Privilege Vector of one Domain allows communication with another, either Domain can initiate that communication. The primary advantage of this feature is that new Domains can be added to a Deployment without requiring that the Privilege Vectors of existing Domains be updated. Access between existing Domains and a new Domain can be allowed by the Privilege Vector of the new Domain. DAC checks are performed at the time an Association is formed.

The Dragonfly Guard provides Confidentiality of User Data. It uses a symmetric key generated using the Fortezza card to encrypt all User Data when it is transmitted between two Dragonfly Guards. The Guard uses the Cipher Block Chaining CBC-64 mode of operation and the Skipjack algorithm on the User Fortezza Card.

The Dragonfly Guard checks for integrity of both User Data and Dragonfly control information when messages are transmitted between two Dragonfly Guards. Messages sent outside of an association are digitally signed. When a message is sent within an association, a checksum is computed and stored in the message before the message is encrypted.

A Security Target provided by ITT Industries describes these security features using the requirements from the Common Criteria for Information Technology Security Evaluation, Version 2. The functionality classes include Audit, User Data Protection, Identification and Authentication, Security Management, Protection of Security Functions, and Trusted Path/Channels. The threats addressed include threats to accountability, confidentiality, integrity of data and software, hardware availability, violation of Mandatory Access Control, and others. (See attached Security Target for complete description.) The User Fortezza card must be configured correctly by the Local Authority, and the user must insert the correct Fortezza card for his environment into the Guard. The configuration is accomplished using a PC Windows-based Administration System that was not evaluated. The Security Target specifies the assurance requirements as Evaluation Assurance Level 2 (EAL2). The Security Evaluation Laboratory of CygnaCom Solutions, Inc. evaluated the Dragonfly Guard against the Security Target as authorized NSA under its Trust Technology Assessment Program. It found that the Dragonfly Guard meets all the requirements of the Security Target and should be awarded a certificate at EAL2. The evaluation was completed September 18, 1998, and the certificate awarded October 1, 1998.

2 Identification

The product described in this Final Evaluation Report is ITT Industries' Dragonfly Guard Model G1.2, running Dragonfly software release 3.0, build 980908.1509. The product consists of an enclosed hardware unit containing a 486 motherboard (with an Intel A80486DX4-100 or equivalent chip), one Ethernet controller built into the motherboard, and one Ethernet Network Interface Card (NIC). The unit's external interfaces include two Personal Computer Memory Card International Association (PCMCIA) card slots, two Ethernet ports labeled local and remote, and a serial port.

In addition, the product received by the purchaser includes an external power supply, an AC power cable, a Fortezza Card (PCMCIA, Type II), a Static RAM Ignition Card (PCMCIA, Type II) containing software, a Dragonfly Guard User's Manual (DF_GUM), and laminated pages called Installation Cards containing installation information.

The software and hardware components of the Dragonfly Guard are further described in section 5. The evaluated configuration, including configuration options, is discussed in section 8. Section 8 also contains a more detailed list of the hardware and software making up the evaluated configuration.

3 Security Policy

This section describes the security policies enforced by the Dragonfly Guard: identification and authentication, mandatory and discretionary access control, and audit. Only the policies are addressed here; the mechanisms that enforce these policies are described in section 5.3, Software Overview.

These security policies are enforced for a Dragonfly Domain. A Dragonfly Domain is defined as the set of hosts that can be directly accessed by a Guard (i.e., without intervening Dragonfly Guards). All hosts within a Dragonfly Domain share the same security level and privileges. Dragonfly Guards do not distinguish (from a security standpoint) among hosts within a Dragonfly Domain.

3.1 Identification and Authentication Policy

Identification and authentication policy includes use of the User Fortezza Card to start up the Dragonfly Guard, and source authentication between Dragonfly Domains.

In order for a Dragonfly Guard to start up, a User Fortezza Card must be inserted. The Dragonfly Guard will cease operating if the User Fortezza Card is removed.

The Dragonfly Guard provides source authentication as a security service. Dragonfly Guards separate two Dragonfly Domains. The Fortezza Card for the Dragonfly Guard contains a User Fortezza Certificate that is used to identify the Dragonfly Guard to other Dragonfly Guards.

Dragonfly Guards establish associations to authenticate each other, to exchange security parameters, and to establish a trusted session for communication. All Dragonfly Messages sent before an association is formed, or outside of an association, are digitally signed. Source authentication is performed by the source Dragonfly Guard signing the Association Request, and the destination Dragonfly Guard verifying the digital signature. Both use the services of the inserted Fortezza cards for digital signature and signature verification.

3.2 Mandatory Access Control Policy

Within a Dragonfly Domain, all hosts are at the same security level. A Dragonfly Guard local to that Domain supports Mandatory Access Control (MAC) by labeling every IP datagram released with the security level of the Dragonfly Guard port on which the packet was received from the originating host. Once the packet is labeled, the security level is compared with the security level of the packet's destination host, as stored in the Host Table during the association establishment process. If this MAC check is passed, further checks are done before the packet is released; see below.

Once the packet is received at the destination Domain, the destination Dragonfly Guard checks the IP datagram's security label against the security level of the destination Domain before releasing the plain text version of the datagram to the destination host.

Dragonfly Guards can support both Write Equal and Write Up. When an association between Dragonfly Guards is initiated, security-related information including the security level of the local Domain is exchanged and stored in the association table. When the security levels of source and

destination Domains match (the Write Equal environment), all IP-based communications are allowed by the MAC policy.

The Dragonfly Guard also allows the transfer of user data from a Domain with a low security label to a Domain with a higher security label (the Write Up environment). Many IP-based protocols require some form of feedback. For example, the File Transfer Protocol (FTP) uses flow control. The feedback constitutes a potential Write Down. How the feedback is handled depends on the protocol.

3.3 Discretionary Access Control Policy

The Dragonfly Guard uses Privilege Vectors for Discretionary Access Control (DAC) between Domains independent of security levels. DAC is checked at the time an association is formed. If the DAC check fails, the association is denied.

If the Privilege Vector of one Domain allows communication with another, either Domain can initiate that communication. This means that new Domains can be added without updating the Privilege Vectors of existing Domains. Access between existing Domains and a new Domain can be allowed by the Privilege Vector of the new Domain.

The Dragonfly Guard's Firewall Mode is an extension of the DAC policy. If a port is configured with Firewall Mode on, all native packets, not processed by another Guard, from a host connected to the port will be discarded, and no native packets will be released to hosts in the Domain connected to the port. (If the hosts are behind another Guard, they are not in the Domain connected to the port.) Protected packets that will be processed by another Guard can be released through the port, subject to MAC and privilege vector checks, and protected packets processed by another Guard can be accepted through the port.

3.4 Audit Policy

A Dragonfly Guard produces audit records depending on the settings of its Audit Mask, which is stored in the Audit Mask certificate signed by the Local Authority on the User Fortezza Card. The Audit Mask is a 256-bit vector with one bit for each auditable event. If an event is to be audited, the bit corresponding to that event is turned on in the Audit Mask. The interface provided by the Administration System lists audit events by name and allows these to be checked as desired. Currently, 23 audit events set by the audit mask are relevant for the evaluated configuration. (See the Security Target for a listing and description of these audit events.) In addition, the startup of the audit function is recorded in the audit trail by the first check-in message from a Guard to its audit catcher and by the first local status message from the audit catcher to its audit trail. If it is configured to generate audit messages, a Guard never stops auditing after startup while it is operating.

Three pre-defined Audit Masks are provided as follows:

- Audit All, which turns on all audit event bits.
- Standard, which means the Guard's Audit Mask is updated from the Audit Catcher when the Audit Catcher's Audit Mask is changed. (Audit Catchers receive audit reports from other Dragonfly Guards and send all audit report messages to their serial port for printing, storage, or subsequent analysis.) The Audit Mask is identified by a version number. The default for the

Standard Audit Mask is all audit events turned on, but the selected audit event bits can be changed by the Administration System.

- Audit None, which turns off all audit event bits.[1]

In addition, the Administration System can be used to define and name new Audit Masks. Any pre-defined Audit Masks can be selected during Guard definition or modified using the Administration System. If the Standard Audit Mask is not selected, only rewriting the card for the Guard will cause modification of the Audit Mask.[2]

Any Dragonfly Guard can be designated an audit catcher. The designation of an audit catcher is done through the Administration System, which is outside of the evaluated configuration. No audit settings can be altered without using the Administration System. Two flags are used by the Administration System. One flag allows the Guard to designate itself as its own audit catcher. The other flag is "Requires audit catcher". If this is set to Yes, the Guard will go into Hold Mode unless it can locate an audit catcher. The Administration System allows up to five audit catchers to be designated for each Guard.

The Guard's User Fortezza Card contains the list of audit catchers for that Dragonfly Guard in the form of a circular list of five entries stored in the Configuration Certificate. When the Dragonfly Guard initializes, it attempts to locate and check-in with an audit catcher on the list, starting with the first entry (designated the primary audit catcher). If the Dragonfly Guard's configuration requires an audit catcher, no user data will be processed until check-in is completed. If an audit catcher is not required, processing of user data will continue whether an audit catcher can be found or not. All audit data from the Dragonfly Guard is sent to the audit catcher, where it is written to the serial port. (From the serial port it can be written to a printer, to a terminal screen, to tape or to other media; however, once it is passed through the serial port it is outside the scope of the evaluated configuration.)

If the current audit catcher goes off-line during Guard operation, the Guard will search down the circular list for the next available audit catcher and use it. If no audit catcher can be located, the Guard will queue audit messages and continue searching for an audit catcher on the list. If the Audit Queue becomes full before an audit catcher is located, the Guard will examine the configuration setting "Requires Audit Catcher". If this is set to Yes, the Guard will go into Hold Mode and stop processing packets until an audit catcher is available. If this is set to No, the Guard will begin sending audit messages to its own serial port.

---

[1] Even if the Audit Mask for the Guard is set to Audit None, the configuration information produced on Guard startup will be written to the Guard's serial port.

[2] Even if the Standard Audit Mask is not selected, the CRL and the Routing Certificate will be updated if the Guard is able to check in with an audit catcher. See section 8.2.1.

If, after the Guard has switched to another audit catcher on the list, the primary audit catcher then comes back on-line, the Guard will continue to use the current audit catcher until that audit catcher goes off-line and the Guard attempts to find another audit catcher on the list. It will not return to the primary audit catcher until that audit catcher is the next on the circular list.

4 Assumptions and Clarification of Scope

The Dragonfly Guard is a special purpose network security product that uses Fortezza Cards to provide Multilevel Secure (MLS) services over a legacy Internet Protocol (IP) based network. The sections that follow describe assumptions made by the evaluation team about secure use of the Dragonfly Guard, and assumptions about the physical environment in which it functions securely. The final section defines the network configurations in which the Dragonfly Guard functions securely.

4.1 Usage Assumptions

In order to provide a baseline for the product during the evaluation effort, certain assumptions about how the product will be used have to be made. The DF_ST (section 3.1) has defined secure usage assumptions, and these were used as a basis for the secure usage assumptions made by the evaluation team. In addition, secure usage assumptions suggested by the vendor's vulnerability analysis (DF_VA) and the evaluation teams' analysis of this document were analyzed and included if warranted.

Assumptions made during the evaluation about the secure use of the Dragonfly Guard are as follows:

Assumption Name	Assumption Description
A.ADMIN	The person acting as the Local Authority is trusted to correctly configure User Fortezza Cards.
A.ATTACK_LEVEL	Attackers are assumed to have a medium level of expertise, resources, and motivation.
A.CRYPTO_SERVICES	Cryptographic services are provided by the User Fortezza Card.
A.CRYPTO_SOF	The cryptographic algorithms on the Fortezza card are assumed to be strong enough to counter at least a medium level of attack.
A.ONLY_PATH	The Guard is assumed to be on the only data path between the two networks connected to its two Ethernet ports.
A.PHYSICAL	The Dragonfly Guard is assumed to be protected from physical tampering.
A.INSTALLER	Authorized installers are assumed to be able to insert the correct User Fortezza Card into the Dragonfly Guard and to connect the correct networks to the local and remote ports.

Table 4-1. Secure Usage Assumptions

4.2 Environmental Assumptions

In order to provide a baseline of the product during the evaluation effort, certain assumptions about the environment in which the product is to be used have to be made. This section documents the two environmental assumptions made about the product during the evaluation.

The Dragonfly Guard is designed to be used between two IP-based networks that may dynamically grow and shrink as the network operates. The evaluation team did consider dynamic network growth as long as a direct wire connection was used. Such dynamic growth is discussed in section 3.3 above.

Direct wire connections were used for all reruns of vendor tests and independent testing by the evaluation team. Vendor testing includes connections to an intranet and to the Internet. Comparisons of actual test results show no differences in test results between direct connections and intranet or Internet connections. Therefore, the evaluation team concludes that connection of the Dragonfly Guard to any IP-based network using 10baseT or 10base2 connections is included in the evaluated configuration.

4.3 Clarification of Scope

This section describes configurations of the Dragonfly Guard and explains what configurations were examined by the evaluation team and which were not. The following sections describe configurations for one and two or more Dragonfly Guards. Next, Dragonfly Guards in more complex configurations are described, followed by use of Dragonfly Guards with Firewalls. Then, Military Network Configurations are discussed.

The threats listed in the DF_ST (section 3.3) are countered as claimed in the DF_ST by the product when used in the evaluated configuration. When the Guard is used in non-evaluated configurations, these threats may or may not be countered.

The Dragonfly Administration System and the Dragonfly Companion are outside of the scope of this evaluation. The Dragonfly Administration System is a software application running on a PC equipped with at least two PCMCIA slots. The network administrator uses the Dragonfly Administration System to specify the Dragonfly Guard or Companion network and security configuration information and then to write this information to the Fortezza Card for that Guard or Companion. That is, the evaluated configuration relies on the Administration System to configure and modify the evaluated configuration's security attributes. The correctness of the configuration can be verified by manually examining the configuration information output by the Guard to its serial port on Guard initialization. (In the DF_ST, ITENV.3 and ITENV.4 document the TOE's dependency on the Administration System. ITENV.1 and ITENV.2 document the TOE's dependency on the Fortezza Card.

The Administration System requires a Local Authority Fortezza Card, provided by ITT, to create valid User Fortezza Cards. A PIN is required to identify the administrator to the Local Authority Card before each use of the Administration System. Although the Administration System is outside of the evaluated configuration, its functionality will be discussed as it applies to the evaluated configuration.

The Dragonfly Companion is a software product that runs on a PC to provide MLS services over legacy IP based networks. It requires a Companion Fortezza Card to identify its user, and is interoperable with the Dragonfly Guard. The Dragonfly Companion will not be further discussed in this document.

4.3.1 Single Dragonfly Guard Between Two Domains

Figure 4-1 shows the simplest configuration of a single Dragonfly Guard between two Domains. All connections shown are direct wire connections. In this configuration, no association between Guards can be formed. Therefore, Guards do not identify and authenticate themselves during association establishment. Only the identification and authentication of the Ignition Card copy of the Fortezza Card PIN takes place when the single Guard initializes. Without an association, encryption keys are not negotiated and no encrypted traffic is passed. Only native mode traffic can be passed, so enabling the Firewall Mode will stop all traffic. Without an association, no privilege vector checking takes place. Therefore, no DAC checking is done. MAC checking is still done, and audit can take place.

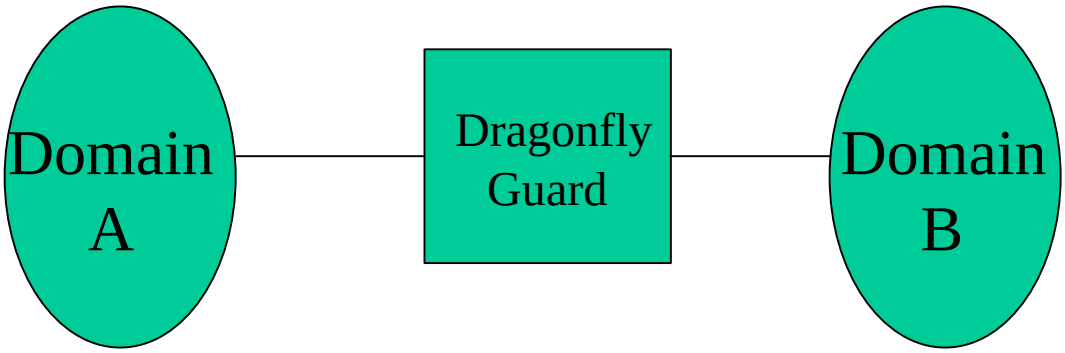

Figure 4-1: Single Dragonfly Guard between two Domains

This configuration has been tested by the vendor and by the evaluation team. This configuration is included in the evaluated configuration.

4.3.2 A Dragonfly Guard for each Domain

Figure 4-2 shows a Dragonfly Guard for each Domain, with a legacy IP-based network between the two Guards. All connections shown are direct wire connections (although connections within the IP-based network cloud can be dial-up as well as direct wire connections). In this configuration, associations between the two Guards can be formed. Identification and authentication can occur between Guards, and encryption can be used to protect packets as they traverse the legacy IP-based network. Privilege vectors can be used to enforce DAC. MAC is checked as described in section 3.2 above. Audit is also performed as required.

This configuration has been tested by the vendor and, in part, by the evaluation team. The configuration used by the evaluation team included a direct wire connection between the two Guards, and did not include a legacy IP-based network. This configuration is included in the evaluated configuration.

4.3.3 A Complex Configuration

Figure 4-3 shows a complex configuration that includes instances of the configurations already shown in Figures 4-1 and 4-2. This configuration was chosen for the evaluation test configuration, because it supports testing of all Dragonfly MAC, DAC, and audit policies.

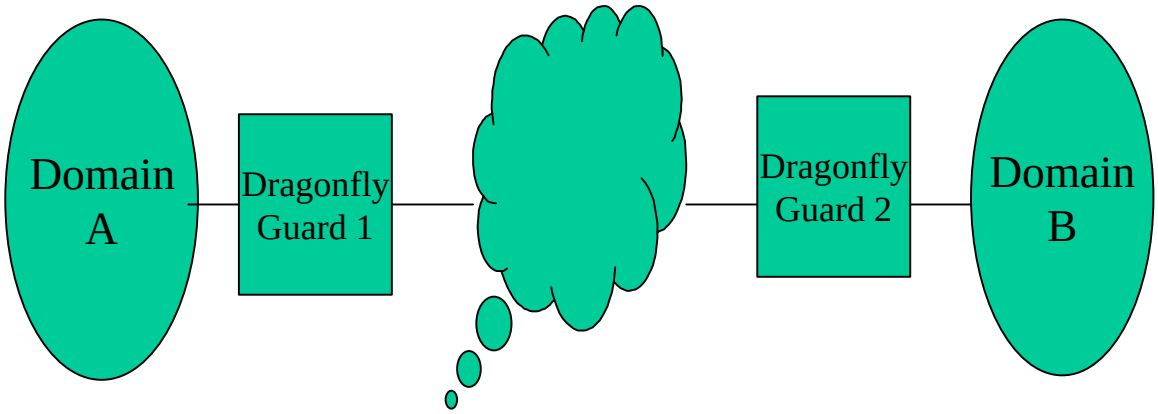

Figure 4-2: Dragonfly Guard for each Domain

Identification and authentication based on the Fortezza Card PIN is always enforced, and MAC is always enforced. However, the enforcement of DAC by privilege vectors, the identification of one Guard to another, and the use of encryption, all of which depend on formation of an association, only occur when two or more Guards are between two Domains attempting to communicate. An association will be formed if a host in Domain B communicates with a host in Domain C, since Guards 2 and 3 are between them. An association will not be formed if a host in Domain C communicates with a host in Domain D.

Figure 4-3 shows additional information. A plus sign next to the Guard number means that write-ups are enabled. Guard 5 has Firewall Mode enabled on the remote port.

This configuration has been tested by the vendor and was used as the test configuration by the evaluation team. This configuration is included in the evaluated configuration.

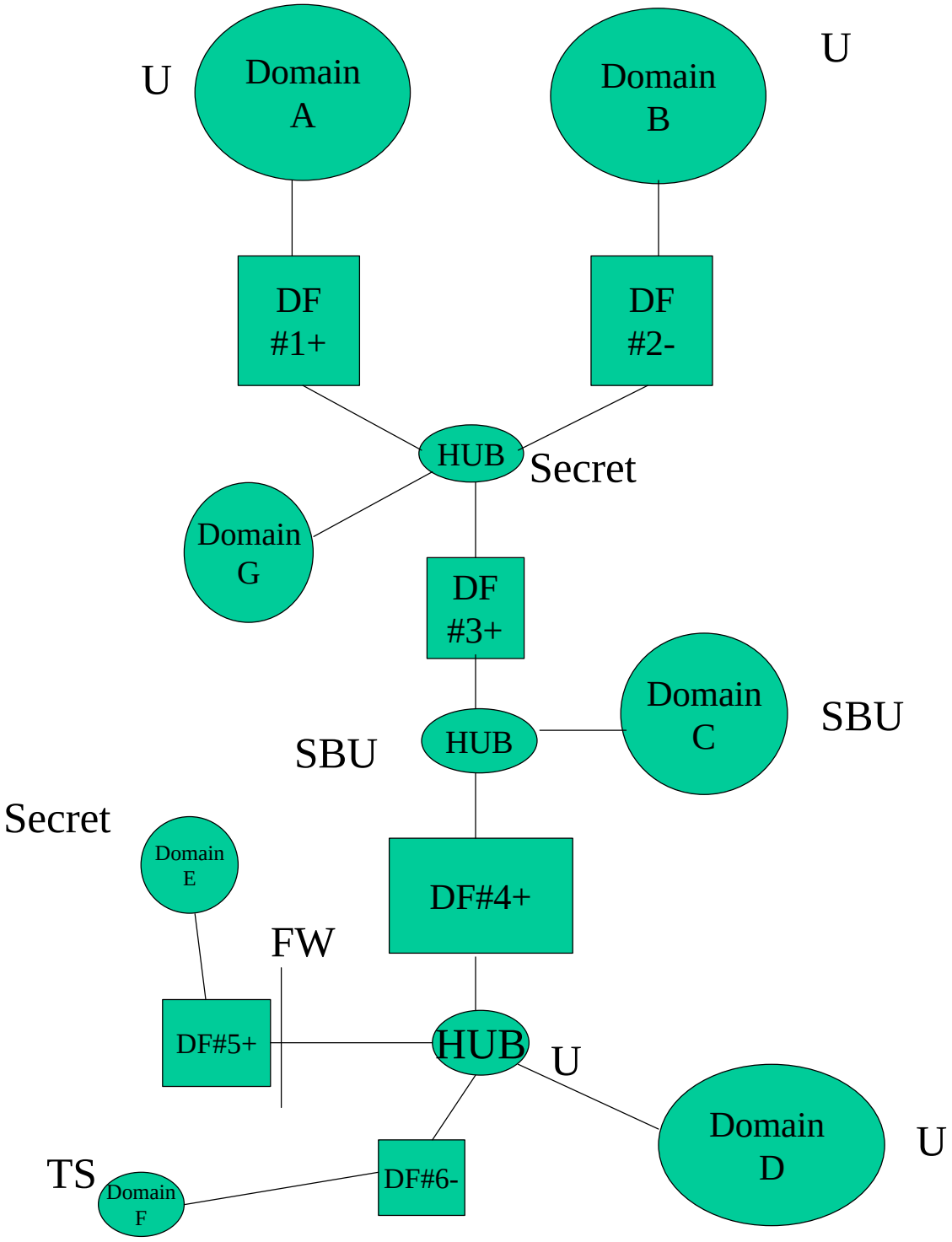
U
Domain A
Domain B
U
DF #1+
DF #2-
HUB
Secret
Domain G
DF #3+
Domain C
SBU
SBU
HUB
Secret
Domain E
DF#4+
FW
DF#5+
HUB
U
Domain D
U
TS
Domain F
DF#6-

Figure 4-3: A complex configuration

4.3.4 Tunneling between Low Domains Through a High Network

A mixed enclave is defined as communication without association. When only one Dragonfly Guard is between two hosts, a mixed enclave may result as shown in the following two examples.

Figure 4-3a shows tunneling by Unclassified hosts through a Secret network. Host 1 is part of Domain 1, which is at the level Unclassified. Host 2 is part of Domain 2, which is also at Unclassified. Both Domains are protected by Dragonfly Guards, and the Guards are connected to a Secret network. Guard 1, which protects Domain 1, is designated as the audit catcher. Both Guards are configured to allow write ups, and Firewall Mode is not on.

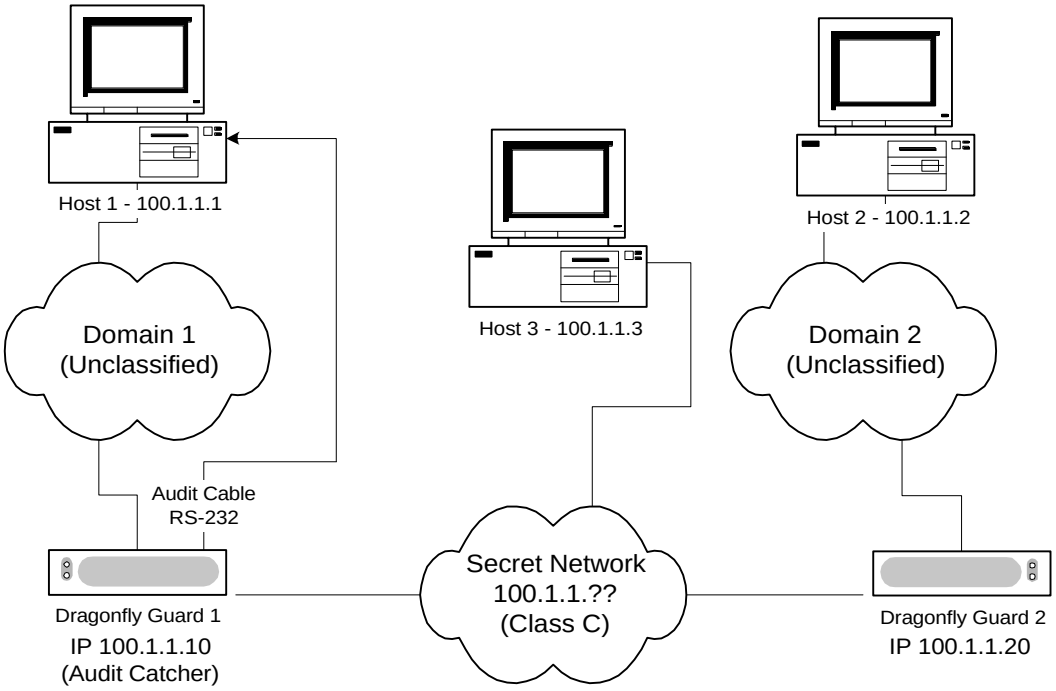

Figure 4-3a: Tunneling through a Secret network

When Host 1 communicates with Host 2, an association is formed between Guard 1 and Guard 2. The association provides encryption between the two Guards, enforces MAC (in this case, a Write Equal situation), and uses privilege vectors to enforce DAC as has been already described for the configuration in Figure 4-2. A tunnel has been built between Host 1 and Host 2 through the Secret network.

Host 3 is connected to the Secret network. When Host 1 contacts Host 3, Guard 1 follows the MAC policy for write-ups. No association is formed, since only one Guard is involved. This is referred to as a mixed enclave. When Host 2 contacts Host 3, the same conditions apply. This configuration has

already been described in Figure 4-1. Enforcement of the MAC write up policy ensures that no user data is allowed to flow from Host 3 to either Host 1 or Host 2.

This configuration has been tested by the vendor and, in part, by the evaluation team, using a subset of the configuration shown in Figure 4-3. The configuration used by the evaluation team included a direct wire connection with hubs between the two Guards, and did not include a legacy IP-based network. This configuration is included in the evaluated configuration.

Figure 4-3b shows a more complex example including a mixed enclave, Domain A. Every host and Guard in Domain A is at the same security level (Unclassified), but Host A is Dragonfly-equipped while Host B is not. Host A and Host B can communicate. Host A and Host C can communicate. Host B and Host C cannot communicate, because the Dragonfly Guard adjacent to Domain A is in Firewall Mode. This configuration is also included in the evaluated configuration.

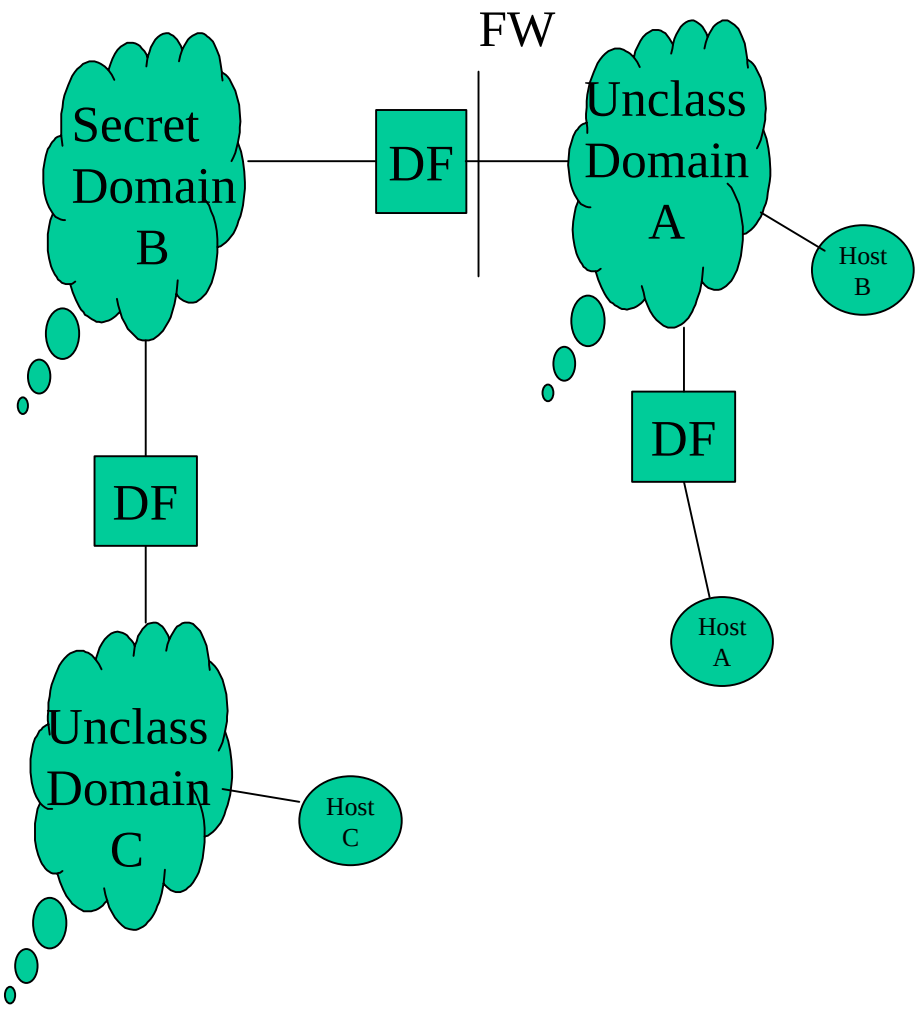

Figure 4-3b Another Mixed Enclave Example

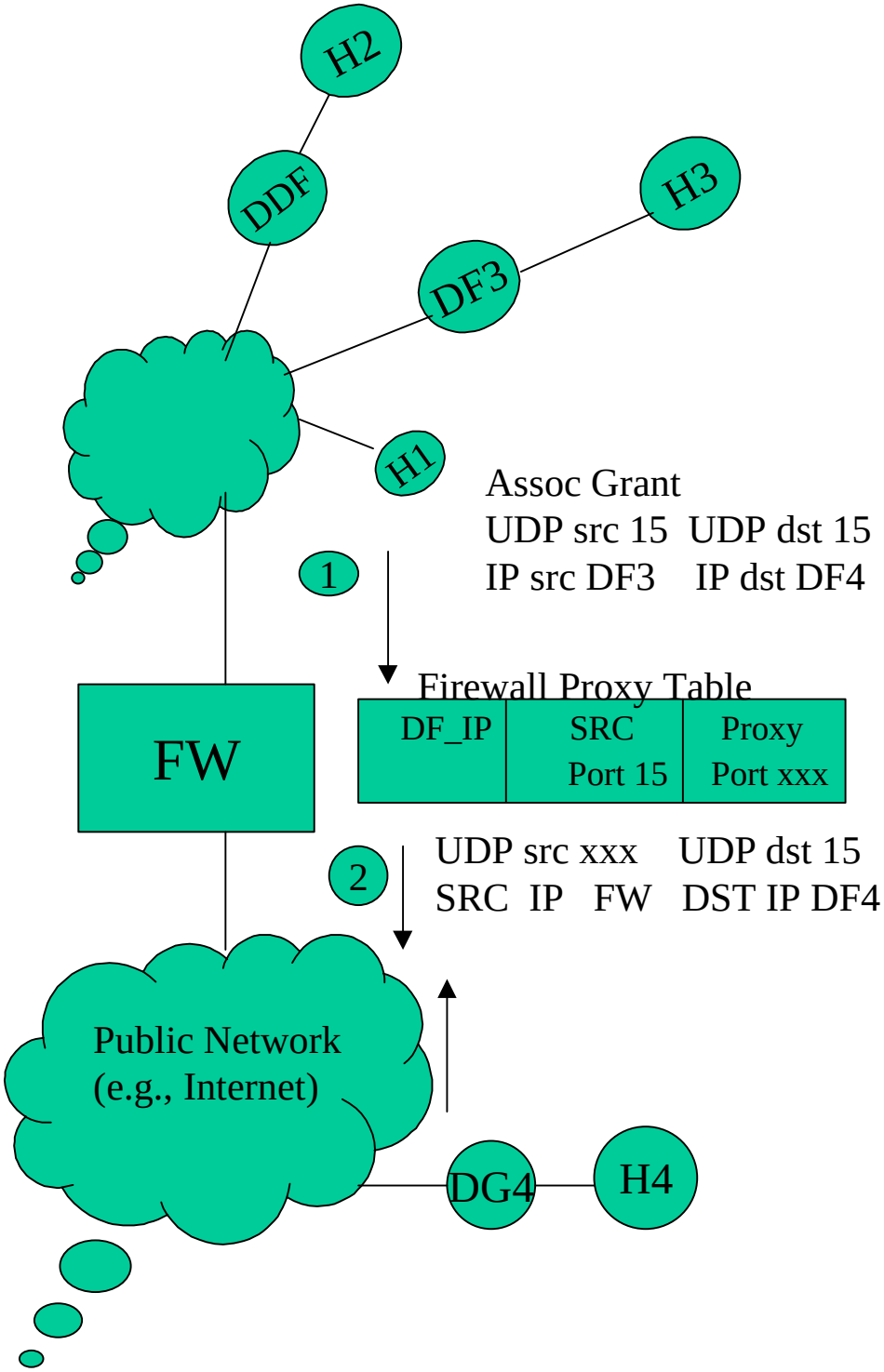

Figure 4-4: Using the Dragonfly Guard with a separate firewall

4.3.5 Using Dragonfly Guards with Firewalls

Figure 4-4 illustrates use of Dragonfly Guards with a firewall. The objective of this configuration is to show how the routing certificate is used to allow Dragonfly protected hosts external to a firewall to have unrestricted, encrypted access to Dragonfly protected Domains inside the firewall. It is assumed that the firewall is initially configured to prevent hosts outside from initiating communications with hosts inside the firewall, and in fact, the IP addresses inside the firewall need not be registered or routable on the external network. This is accomplished through the use of a routing certificate and a Dragonfly Guard referred to as the Designated Dragonfly or DDF.

4.3.6 Military Network Configuration Issues

The Dragonfly Guard is designed to support Department of Defense (DoD) networks such as the Secret IP Router Network (SIPRNET) and the Unclassified IP Router Network (NIPRNET). The evaluation team did not look at connections to either network, or to any other DoD network, as part of the evaluation.

5 Architecture

5.1 System Overview

The Dragonfly Guard is a network security device that uses proprietary hardware and software to provide a security boundary between two adjacent Dragonfly Domains. The Dragonfly Guard uses National Security Agency (NSA) Fortezza Cards to provide multi-level secure (MLS) services to legacy networks, that is, Internet Protocol (IP) networks that operate in System High mode. The Dragonfly Guard also serves as an in-line encryptor. Dragonfly Guards protect enclaves or individual hosts. Within a network, Dragonfly Guards are in-line between hosts and the network. Dragonfly Guards operate on standard IP datagrams.

A Dragonfly Guard is an enclosed unit containing a 486 motherboard and two Ethernet Network Interfaces. The unit has two Personal Computer Memory Card International Association (PCMCIA) card slots, two Ethernet ports labeled local and remote, and a serial port.

Dragonfly Guards require two PCMCIA cards to operate. The first card is the Ignition Card that contains the Dragonfly software and is digitally signed. The second card is the User Fortezza Card that contains the configuration information for that particular Dragonfly Guard. The User Fortezza Card contains eight certificates. Five of them, the User, Configuration, Audit, the Certificate Revocation, and the Routing certificates, contain configuration information and are signed by the Local Authority. The other three are the Local Authority, the root, and the root authority certificates, which form a certificate hierarchy where each certificate in the chain is signed by the private key associated with the public key in the certificate above it. The Dragonfly Guard uses the Fortezza card for hashing, digital signatures, key generation, and encryption.

A Dragonfly Guard separates two Dragonfly Domains for MAC, with one Domain attached to each port. A Dragonfly Domain is a set of computers that are networked together without any intervening Dragonfly Guards. These computers in the same Domain may be PCs, Workstations, or Servers that are all at the same security level. The two ports are labeled remote and local for convenience, although processing is actually the same whether the port is local or remote.[3]

5.2 Hardware Overview

As shown in Figure 5-1, the Dragonfly Guard is constructed of the following subsystems:

5.2.1 CPU Board

The CPU board provides PC-compatible processing, memory, two serial interfaces, one Ethernet interface, and a PC/104 bus for interfacing to other hardware subsystems. Comm1 is the Audit Output port (see 5.2.7 below) and Comm2 is used to drive the front panel indicator lights.

[3] As described in section 4, the full security functionality provided by the Dragonfly Guard requires formation of an association between two Dragonfly Guards.

5.2.2 Flash Floppy

The Flash Floppy is a 1.5 MB bootable disk that is implemented with flash memory chips. It is the only non-volatile storage in the Dragonfly Guard. It is used to store the operating system software, and the Software Load and Validation Modules.

5.2.3 Ethernet Interface

The two Ethernet subsystems provide both a 10base2 and 10baseT interface for the Dragonfly Guard's Local and Remote ports. As Figure 5-1 shows, memory for the Local port is accessible by the Local Ethernet Controller and by the CPU, but not by the Remote port, while memory for the Remote port is accessible by the Remote Ethernet Controller and by the CPU but not by the Local Ethernet Controller. Although Figure 5-1 shows the local Ethernet as a separate unit, it is integrated on the CPU board. The remote Ethernet is a PC/104 Card.

5.2.4 PCMCIA Reader

The PCMCIA Reader provides a mapping of both the Fortezza and Ignition Card contents into the address space of the CPU. This allows the CPU to access memory and control registers on both cards.

5.2.5 Fortezza Card

The Fortezza Card provides all cryptographic services used by the Dragonfly Guard. The Fortezza Card implements SKIPJACK symmetric encryption and decryption, Secure Hash, Digital Signature, Key Generation, and Key Exchange algorithms.

5.2.6 Ignition Card

The Ignition Card stores the main Dragonfly executable (as opposed to DOS and the startup software, which is stored on the flash memory card on the motherboard). The contents of the Ignition Card are signed by the Dragonfly Software Authority's private key (see DF_CM). During initialization, this signature is checked. A Dragonfly Guard will not unpack the contents of an improperly signed Ignition Card. The date of the software build image on the Ignition Card is also written in human-readable form on the outside of the Ignition Card.

5.2.7 RS-232 Port

In the evaluated configuration, the serial port supports output of audit data only. (In non-evaluated configurations of the Dragonfly Guard, the serial port can support PPP or SLIP two-way connections.)

5.2.8 Power Supply

Dragonfly operates on 5 volts DC @ 2 amps typically, as provided by the power supply.

5.2.9 Case

The Dragonfly Guard case is constructed of heavy gauge extruded aluminum, and provides the external physical interfaces. These include two 10base2 and 10baseT Ethernet port connections, one marked “Local” and the other marked “Remote”; a DB-9 serial port marked “Audit”, a power connector, and two LED’s, one green and one red (Run = green, Error = red).

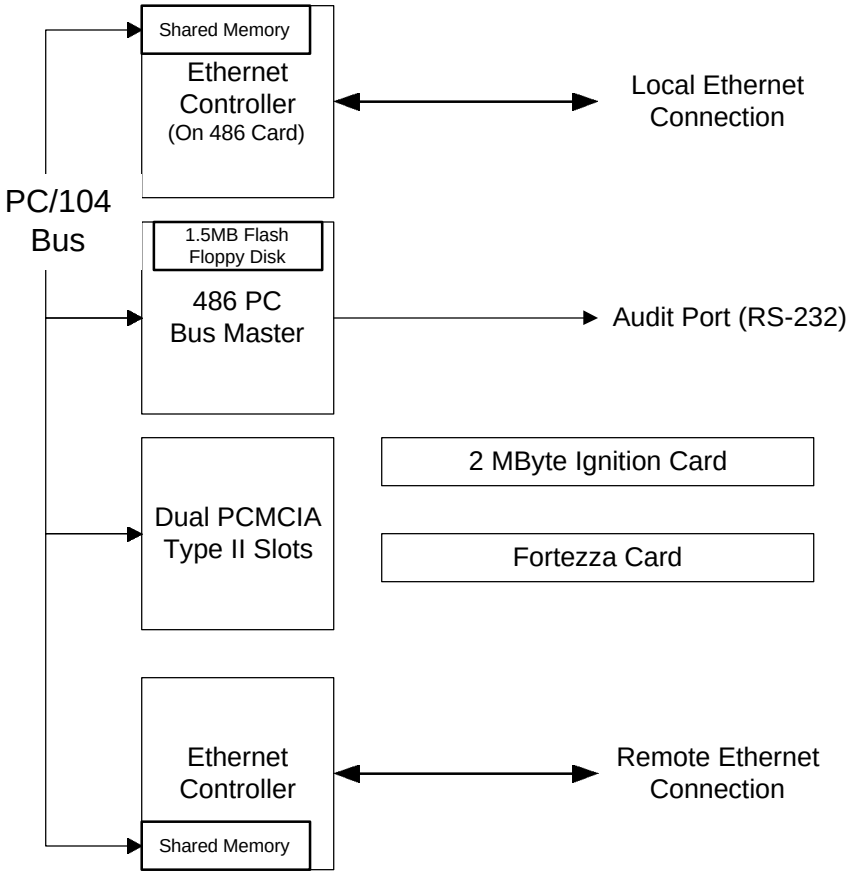

Figure 5-1: Dragonfly Hardware Architecture

5.3 Software Overview

The Dragonfly Guard software is stored on the ignition card inserted into one of the PCMCIA slots and the flash memory disk inside the system unit. Dragonfly Guard software is written in Borland C with some inline assembly code, but is shipped only as executable code on the Ignition Card or on the flash memory disk on the CPU board. The flash memory disk includes MS DOS Version 6.2 and files for Dragonfly software validation and load. The software loaded from the Ignition Card supports both Dragonfly initialization and Dragonfly operation.

The Dragonfly Guard software is divided into the following software subsystems:

5.3.1 Boot and Self Test

After the Guard is powered on, standard CPU and memory diagnostics, built into the motherboard, are run, and success or failure is signaled with a series of beeps. If the tests are successful, the boot process continues with the loading of MS-DOS Version 6.2, which reads config.sys and configures its environment accordingly.

The use of MS-DOS by the Dragonfly Guard is minimal. There is no external interface provided to MS-DOS by the Dragonfly Guard. The serial port is configured in the evaluated configuration to allow output of printable ASCII characters only, and there is no network stack loaded that could provide access via the Ethernet ports. Thus, there is no external interface to allow exploitation of vulnerabilities that might exist in MS-DOS. More importantly, once the Dragonfly code is loaded, it controls the processor, as MS-DOS is a single user, non-preemptive operating system. The only calls to MS-DOS by the Dragonfly software are to set and obtain time values.

5.3.2 Software Load and Validation

At the end of the MS-DOS boot process, autoexec.bat is read, which causes the following two modules to be executed:

- Dpcmcia.exe maps the Ignition and Fortezza card contents into the processor's memory space.
- Verifile.exe checks the digital signature obtained from the software on the Ignition Card against the Dragonfly Software Authority public key contained in the validate.exe software. If the digital signature validates, then the files are unpacked onto a RAM disk and startup.bat is executed. At this point, the Ignition Card is not accessed again, and can be removed.

5.3.3 Dragonfly Initialization

The initialization is accomplished in several steps. First, setport.exe is run to read and validate the Configuration Certificate from the Fortezza Card. The Packet Drivers for the Ethernet cards are loaded.[4] Next, Dfly.exe, the Dragonfly executable, starts and does the following:

- Allocates memory. All memory is allocated at startup and is static. The four main memory heaps are the local port untrusted packet memory, the remote port untrusted memory, the trusted packet memory, and the trusted carrier memory.
- Initializes the IP protocol stacks for the local and remote ports.
- Initializes the Fortezza Card, and reads and validates all certificates using the Secure Hash and digital signature functions of the Fortezza Card. This is the second time the Fortezza Card has been initialized.
- Writes the Dragonfly Guard configuration data to the serial port for verification purposes.

[4] At this point, if this is not the evaluated configuration and PPP is configured, commands would be written to the serial port to command the modem to dial out. In the evaluated configuration, no software is loaded to the serial port to support interactive communications. Instead, the serial port is used to output audit information.

- Sets the CPU time to match the Fortezza Card's clock.
- Starts the main event loop.
- Checks in with the audit catcher.

5.3.4 Dragonfly Operation

The main event loop polls for data objects that need servicing. Three interrupt-driven tasks can occur; these are local port Ethernet packet processing, remote port Ethernet processing, or timer interrupt. The local and remote Ethernet packet processing transfers packet data to and from packets allocated to the appropriate heaps. When a packet is received from the network, the interrupt processing obtains an empty packet from the appropriate untrusted packet memory heap, copies the received packet into it, and places it on the appropriate received packet list as follows:

- Local port network software only deals with packets from the Local port packet memory.
- Remote port network software only deals with packets from the Remote port packet memory.
- The main event loop, as the Trusted Core, has access to all memory.

The processing of data objects implements the security policies described in section 3. The main event loop, as the Trusted Core, makes all decisions on what associations are to be established, and what packets are to be transmitted. The paragraphs below describe association establishment, followed by a description of Trusted Core main loop packet processing.

Trusted Core main loop packet processing, once initiated, continues forever, performing the following tasks as needed:

- Receive Packet Interrupt Processing
- Input Packet Processing
- Fortezza Processing

If more than two Dragonfly Guards are between the hosts wishing to communicate, an extra step is added to the association establishment process that allows symmetric keys to be negotiated for use between the pairs of Guards involved. The Guard in the middle is known as the adjacent Guard. These symmetric keys are known as release keys. The association request will include two certificates, one for the originating Guard and one for the adjacent Guard. The association reply will contain, in addition to the fields required to negotiate the symmetric keys for the association, the fields required to negotiate the release keys. The process for determining release keys is repeated until all pairs of Guards between the originating and destination Guard have negotiated release keys.

6 Documentation

This section describes the documentation provided with the Dragonfly Guard by ITT Industries. This is a User Manual, which is addressed to the installer of a single Dragonfly Guard, and a set of laminated Installation Cards. This documentation describes the secure setup, installation, and operation of the Dragonfly Guard, but does not give directions for modification of security attributes. The Dragonfly Administration User Manual, which covers the administrative system and is needed to modify security attributes, is described in Annex A since the Administration System is not part of the evaluated configuration. (It does not come with each Guard.)

Dragonfly Guard documentation version numbers apply to the version of the document itself and do not necessarily reflect the Guard model number or the software release number. Also, the document itself may not reference the exact release of the software to which it applies.

6.1 Dragonfly Guard User Manual

The Dragonfly Guard User Manual (DF_GUM), Version 2.02, dated August 1998, provides installer guidance. A installer is defined here as the person responsible for setting up a Dragonfly Guard at its location, inserting the Ignition and Fortezza cards, and maintaining the connections between the Dragonfly Guard and the local and remote networks. This installer does not have access to the Administrative System that writes Fortezza cards, and is not responsible for network administration.

The DF_GUM is divided into two sections. Section I describes the functions and interfaces available to installers of the Dragonfly Guard, including the installation, operation, and trouble shooting of a Dragonfly Guard unit. The installer-accessible security functions described here include the audit connection, the insertion of the correct Fortezza and Ignition cards, and maintenance of correct network connections.

Warnings about what must be controlled in a secure processing environment are noted where appropriate. These warnings present installer responsibilities needed for secure operation of the Dragonfly Guard unit.

The secure usage assumptions found in the DF_ST that apply to installer behavior or items under installer control include:

A.ONLY_PATH	The Guard is assumed to be on the only data path between the two networks connected to its two Ethernet ports.
A.PHYSICAL	The Dragonfly Guard is assumed to be protected from physical tampering.
A.INSTALLER	Authorized installers are assumed to be able to insert the correct User Fortezza Card into the Dragonfly Guard and to connect the correct networks to the local and remote ports.

The DF_GUM describes all security requirements for the IT environment that are relevant to the installer, including physical control of the Fortezza and Ignition cards, physical control of network connections, correct installation of components, protection of audit output, and procedures to verify correct operation. These descriptions address the three secure usage assumptions listed above.

Section II of the DF_GUM is identical to Section II of DF_AUM. Section II provides an overview of the Dragonfly Guard, a description of using the audit catcher, a discussion of co-existence between firewalls and Dragonfly Guards, and a discussion on configuring Dragonfly Networks. Automatic discovery by which Dragonfly Guards learn about each other is described, and different configurations are covered. The output of the Audit Catcher and how to read it is described. The audit event numbers are mapped to event names and descriptions. The revocation of User Certificates is covered.

The information in both sections has been found to be consistent with the information in other Dragonfly documents furnished to the evaluators.

6.2 Installation Cards

The Installation Cards delivered with the evaluated configuration consist of several laminated cards bound together by a plastic ring. The Installation Cards are not identified by version number or date. On the cards are installation and setup directions, divided into steps, and illustrated with pictures of the Guard and of each accessory as it is discussed. The information is the same as that provided in the DF_GUM, section 2, but is presented in a simplified, detailed fashion. The purpose of the cards is to guide untrained personnel, typically soldiers, through proper setup and installation of the Dragonfly Guard to the point where the Guard can begin operating securely.

7 Product Testing

This section describes the testing performed as part of the evaluation. It includes the team's analysis of the vendor test documentation, the re-running of the vendor's test suite, and the running of the team's own security tests. First, the analysis of the vendor's testing effort is given followed by a description of the evaluation team's testing.

7.1 Analysis of Vendor's Testing Effort

The vendor's testing effort is described in DF_TPROC. This document includes the test plans, the test procedure descriptions, and the expected test results. In addition, the vendor provided a complete set of actual test results for each test procedure. The paragraphs that follow describe the test suite in more detail, the test configuration, the test coverage and depth analysis, the testing approach, and the results of vendor testing.

7.1.1 Details of Test Suite

The test plans included in DF_TPROC are informal descriptions of the test procedures that follow, and indicate what security functionality is being tested. The test procedures themselves are in the form of steps. For each step, "tests" are defined to indicate each expected result for that step. For example, step 8 of Table 1-4 is "Ping from A to D". Test A is for the audit catcher to report audit event code 8 (association deleted). Test B is for the audit catcher to report audit event code 12 (no association entry). Test C is for the audit catcher to report formation of a new association. Test D is for the ping to be received at D. For each "test", the result column of the test procedure allows Pass (expected results obtained) or Fail to be circled. If all "tests" pass, then the step is passed.

Section 1.1 of DF_TPROC describes the setup for the test configuration, including security levels, audit configurations, privilege vectors, Firewall Mode settings, enabling write-ups, key expiration period, and association time out periods. The configuration of the User Fortezza Certificates, Guard configuration certificates, Audit Masks, and CRLs requires use of the Administration System, which is outside of the evaluated configuration.

Section 1.2 describes the test generation of audit events to cover all audit codes documented in DF_GUM and the design documentation. Table 1-2 gives references to other tests where the audit event is generated when possible, or describes how the audit event is generated for this test.

Section 1.3 gives the access control tests for *ping*s and FTP PUTS and GETS in Tables 1-3a and 1-3b. These tests check out every combination of DAC, MAC, and other configuration settings allowed by the test configuration setup. Table 1-3c requires changes to security levels on selected ports. Table 1-3d reruns certain tests from Tables 1-3a and 1-3b with the changed configuration, showing that the security level changes are in fact effective.

Section 1.4 contains Guard-specific tests to exercise audit catcher required, Firewall Mode, and other configuration settings. Table 1-3 in this section lists 47 steps covering association establishment and

audit, association recovery, search for alternate audit catcher when primary audit catcher fails, updates of Audit Mask and CRLs from the audit catcher, termination of associations with Guards with User certificates on the CRL, use of privilege vectors, expiration of associations based on association time to live parameter, fault upon removal of the Fortezza Card from a Guard, and various integrity checks. Some of the test steps require use of NetXray to capture, change, and resend messages.

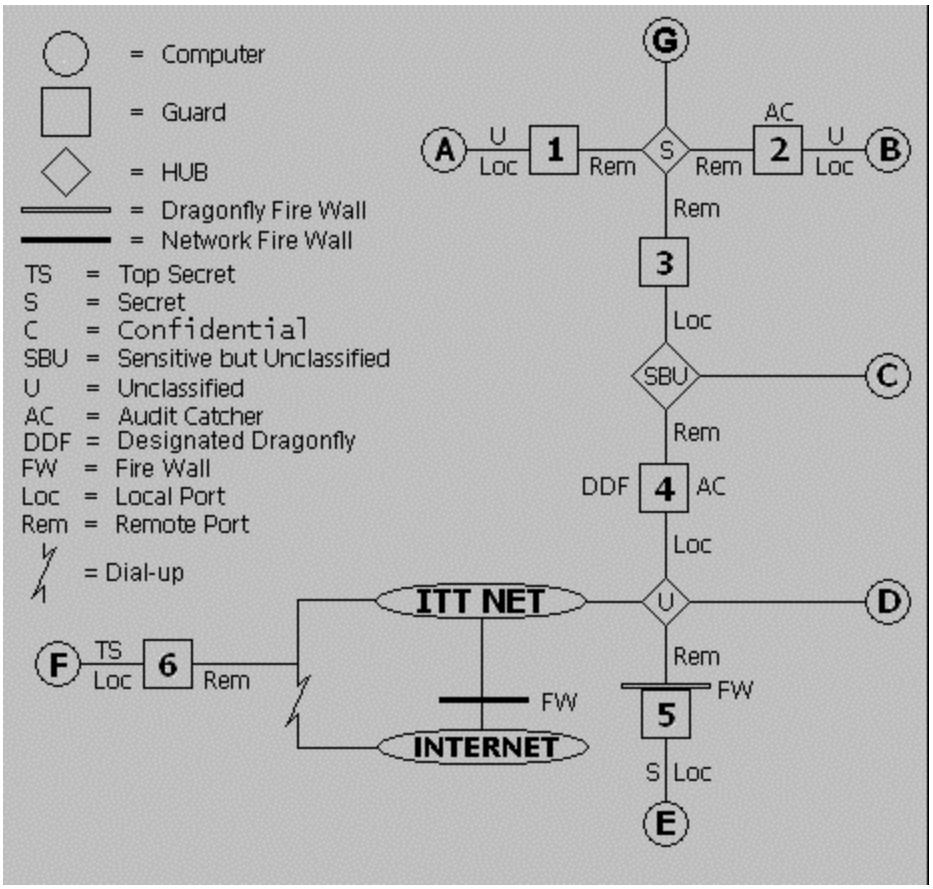

Figure 7-1: Vendor's Test Configuration

7.1.2 Test Configuration

The test configuration used by the vendor is described exactly by a configuration drawing (see Figure 7-1) and Table 1-1 of the DF_TPROC, which describes how to configure each of the six Dragonfly Guards used in the test configuration. Figure 7-1 gives the security levels and connections for each Dragonfly Domain (represented by a single host), each Dragonfly Guard, and each single-level hub used in the test configuration. Domains A through G are configured for the tests, connected to the six Dragonfly Guards and the three hubs as shown in Figure 7-1.

In addition to the Dragonfly software contained in the Dragonfly Guard units and the Ignition Cards, the following software was used by the vendor to support testing:

- NetXray, to format and alter packets as part of the tests.
- Dragonfly Administration System, to allow alteration of Fortezza Card information.

7.1.3 Coverage and Depth Analysis

Test coverage analysis was based on the vendor's claims in DF_CD. This document maps each security function to the portion of the DF_HLD that describes it, and then to the test procedure that verifies that function. The DF_CD also maps the Dragonfly Guard external interfaces to the DF_HLD references, but does not include test procedures. In addition, the evaluation team compared the test plan and test procedure description for the test to the claims made in the DF_CD. Also, the actual test results were consulted when necessary.

The team's test coverage analysis showed that the test procedures did not include tests for altered or incorrect anticipated responses. This was discussed with the vendor, and the test procedures were augmented. The vendor also added tests for audit generation, the Confidential security level, and change of MAC security level based on the evaluation team's test coverage analysis.

7.1.4 Testing Approach

The vendor's testing approach is black box testing concentrating on claimed security functionality and using the Guard's external interfaces. The test suite serves as regression tests to verify the proper operation of new versions of the Guard's hardware and software. This test suite does not include any of the module level tests used during software development and integration.

The first tests show that the configuration options are working as expected, by trying every combination of Domains provided in the test configuration. The tests proceed by showing the effect of a configuration setting, changing the configuration setting, showing that the test result has changed as expected to match the new configuration setting, and examining the audit records from the test. The anticipated message tests capture traffic from an allowed write up, then alter that traffic to show that alterations to user data – that is, data copied from high to low messages and inserted into the anticipated response – are not allowed.

7.1.5 Results of Vendor Testing

The vendor provided a complete set of test results for all of the test procedures based on software release 2.04, build 980825.0035, showing that all documented tests in the DF_TPROC had been successfully run. The vendor later provided actual test results for all tests of software release 3.0, build 980908.1509. The only expected change was the existence of a 32-bit checksum in protected user datagrams from release 3.0 instead of the 16-bit checksum used by release 2.04. This difference was confirmed by examining packets displayed by the NetXray network sniffing tool. The actual test results comprise an annotated copy of the test table, with handwritten notes as to outcome and any unexpected behavior; files from the audit catcher capturing Guard initialization and all audit events for the test for each Guard involved; and NetXray snoop output when part of the test. The evaluators confirmed that the expected results were obtained, or that corrective action was taken when problems were found.

Some problems were uncovered by the vendor during production of these actual test results, and these problems and their fixes were documented with the actual results. The evaluation team analyzed these results and confirmed that all tests had been run. By the end of the test period, the evaluators confirmed that the expected results matched the actual results for all test procedures.

7.2 Evaluation Testing

The evaluation team's testing approach had two parts. First, the team concentrated on reproducing and confirming the vendor's test results. Next, the team conducted independent tests.

7.2.1 Test Configurations

Four configurations were used by the evaluation team's testing effort. With the vendor's help, the evaluation team reproduced the vendor's test configuration in the evaluation lab as exactly as possible. Figure 4-3 shows the evaluation team's test configuration. For this configuration, each Domain contained one host.

Figure 4-3 is similar to Figure 7-1, from the vendor's test documentation (DF_TPROC). The outside connections to the Internet and to ITT Net shown in Figure 7-1 for Domain F were not reproduced. Instead, all of the Guards, hubs, and hosts were directly connected. The Internet connection is shown in Figure 7-1 as dial-up, which is outside of the evaluated configuration. The design of the Guard is such that the actions of the Guard do not depend on whether the Domains are directly connected or connected through a network such as ITTnet. The evaluation team confirmed that the results obtained from their reruns matched the actual test results obtained by the vendor exactly, even though the connections for the reruns were directly connected and not through a network.

During later test periods, a configuration corresponding to Figure 4-1 was used for the anticipated message tests. These were tests the vendor added in response to evaluation team suggestions. This was a subset of the larger test configuration, consisting of Guard 4 and Domains C and D. Each Domain contained a target host and another PC running NetXray to capture message traffic. Domain C was at SBU and Domain D was at Unclassified.

The independent tests used two different configurations. The usual test configuration involved two Guards and two Domains at different levels with a laptop running NetXray in between to capture and modify packets as required.

Finally, for scanner tests of Firewall Mode, a configuration was used that corresponded to Figure 4-1 but with the Dragonfly Guard connected to a host running the scanner tool against the Guard protecting one host.

7.2.2 Rerunning Vendor Tests

The evaluation team had several goals in choosing vendor tests to rerun. First, the team wished to show that the team's test configuration was setup and operating as expected, and that the documented expected results were obtained from the test configuration. This was during the first test

period, before actual test results were available. During this period, the evaluators confirmed that the Guard output its configuration information to the audit catcher on startup, that performing certain actions produced expected audit messages, that check-ins with the audit catcher occurred as documented, and that changes to the configuration on the Fortezza Card by the Administration System would be shown on Guard startup and verified by changed behavior during the test. The evaluation team also became familiar with the use of NetXray to capture, manipulate, and resend messages.

During this initial stage, the evaluation team began with the test configuration setup corresponding to Figure 4-3. This required use of the Administration System, which is not part of the evaluated configuration. The team, augmented by a vendor participant, then reran Table 1-2 from the vendor's test documentation (DF_TPROC). At a later date, the evaluation team, working without the vendor and with a subset of Domains B, C, D, and E present, reran a subset of Table 1-3 in section 1.2 of DF_TPROC. FTP servers were installed on all hosts for the second round of vendor participation, and tests in Table 1-3 were rerun then. Results were as expected unless a configuration error had been made during preparation of the Fortezza Card. We reran the tests, correcting the Fortezza Card, until the expected results were obtained.

Working without the vendor, the team ran steps 1-9 of Table 1-4 in section 1.3 of DF_TROC with all six Guards configured according to Table 1-1 but only hosts for A (connected to Guard 1's unclassified local port) and D (connected to Guard 4's unclassified local port). These steps verify audit reporting during association establishment, verify proper duration of associations, and verify the proper recovery of associations when Guards are rebooted. With the vendor participant present, the team reran steps 9-20 and 44-47, reconfiguring Guards 4 and 5 and using only hosts A, B (connected to Guard 2's unclassified local port), and C (connected to Guard 3's local SBU port). Steps 9-20 verify that audit masks and CRLs are properly distributed by the audit catchers, that a Guard that requires an audit catcher will go into Hold Mode if one is not available, that a Guard will search for an alternate audit catcher if its primary one goes off-line, that associations are allowed or denied based on privilege vectors, and that associations time out when the inactivity period exceeds the configured “time to live”. Steps 44-47 verify that Guard software is authenticated upon startup, and that integrity checks are performed on digitally signed messages such as Association Requests, certificates, and user data packets. All results matched the vendor’s expected results. These steps (especially steps 44-47) were chosen because they appeared frequently in DF_CD as showing security functionality.

During the next stage, the evaluation team chose a subset of tests to rerun in order to better understand the policy for allowing Write Ups when that setting was enabled. Working with the vendor, the team reran anticipated message tests in Tables 1-7 (ARP), 1-8 (ICMP), and 1-9 (FTP) from DF_TPROC. The evaluation team looked at the captured traffic in detail to confirm design details in addition to confirming the expected results. Because NetXray allowed the evaluation team to examine the messages, a deeper understanding of the MAC policy was obtained. We did not have SMTP or DNS servers on any hosts in the test configuration and so could not rerun those vendor tests, but analysis of the actual test results showed corresponding results for SMTP. (For DNS, we ran an independent test to examine the request message; see below.)

The evaluation team executed nearly all of the vendor's test suite. The only significant test set that was not duplicated were the tests from Table 1-6, which verified how the Guard works with a firewall product as a Designated Dragonfly. (See Section 4.3.5.) These tests were not run because this was not an evaluated feature. The last stage of testing was that of independent tests, as described below.

7.2.3 Independent Tests

The evaluation team supplemented the vendor test suite showing proper handling of input with negative tests to show that errors and unexpected input were handled correctly. In addition, claims made by the design documentation that had not been clearly shown by the rerun of the vendor tests or the actual test results provided were further explored. The test configuration usually involved two Guards and two Domains at different levels with a laptop running NetXray in between to capture and modify packets as required.

The independent tests conducted by the team, with vendor participation and assistance, can be summarized as follows:

- FTP tests to try all commands listed as allowed and some listed as not allowed. These commands are below the level of the user interface. Use of NetXray to analyze the traffic allowed the underlying commands to be identified. Some unexpected results were obtained and passed to the vendor for explanation, leading to updates in Table 1-9, step 9, and changes such as correct documentation of the audit event for the test.
- Insertion of wrong Fortezza and bad Ignition Cards. First, we inserted the Local Authority Card from the Administration System into a Guard. The initialization process failed with the expected error. Next, we inserted an Ignition Card with no digital signature on the software, and again the initialization process failed with the expected (different) error.
- Forcing expiration of a User Certificate on a Fortezza Card. With the expired certificate, the Guard would not initialize.
- Examination of Guard initialization output to confirm configuration changes.
- Examination of what happens when a Guard has a later CRL than its audit catcher does. (It generates an audit event, "Old CRL Version".)

Finally, the evaluation team ran the commercial tool, Internet Scanner, version 5.2, against the Dragonfly Guard in Firewall Mode. The test configuration involved a computer running the tool connected to the remote port of a Guard and a host on the local port, with the local port configured in Firewall Mode. This tool probes an IP-addressed host for over two hundred well-known vulnerabilities in IP, UDP, TCP, and UDP- and TCP-based applications like SMTP and FTP. There was only one vulnerability found, that the Guard responded to traceroute. This was identified as a low risk vulnerability by the tool. Response to traceroute is not configurable on the Guard and is usually considered acceptable when firewall products are configured for actual use.

8 Evaluated Configuration

This section describes the evaluated configuration of the Dragonfly Guard. The purpose of an evaluated configuration is to describe what hardware and software components of a product were examined by an evaluation team, given the personnel and environmental assumptions described in section 4, Assumptions and Clarification of Scope. It also provides advice and guidance on how to configure and use the evaluated product in a secure manner.

The information in this section can be used by security architects in constructing network architectures based on the Dragonfly Guard. However, a threat and risk assessment must be performed on the network in question to support secure use of the Dragonfly Guard.

8.1 Evaluated Hardware and Software Components

The following information reflects the exact configuration examined by the evaluation team to produce the results documented in this report. This configuration information was obtained from the configuration list given in section 2 of DF_GUM, as amplified by information in DF_CM. In addition to the hardware and software documented here, the user also received documentation with the evaluated configuration as defined in section 6.

8.1.1 Evaluated Hardware Components

The Dragonfly Guard is Model G1.2, which consists of an enclosed hardware unit constructed of heavy gauge extruded aluminum. The model number is found on a label on the bottom panel. The hardware unit contains a 486 CPU board with one of the following chips or equivalent:

- AMD A80486DX4-100NV8T
- AMD A80486DX4-100SV8B
- Intel A80486DX4-100

The BIOS is American MegaTrends Incorporated 486 PCI ISA. The CPU board includes two serial interfaces, one Ethernet interface (for the Local Port), and a PC/104 bus, which supports the PC/104 Ethernet card (for the Remote Port) and the Personal Computer Memory Card International Association (PCMCIA) reader.

The CPU board contains non-volatile storage in the form of a 1.5 MB bootable “disk” (non-volatile, “flash” memory) configured as drive a: and called the Flash Floppy Drive. It is used to store the MS-DOS and the Dragonfly-specific startup routines.

The Guard unit has the following external interfaces:

- Two PCMCIA Type II slots
- Two Ethernet ports, each supporting 10base2 and 10baseT Ethernet port connections, one marked Local and the other marked Remote
- A DB-9 serial port (supported by Comm1 on the CPU board)
- Two indicator lights, one red and one green (supported by Comm2 on the CPU Board), and
- A power connector.

In addition to the Guard unit, the following equipment comes as part of the evaluated configuration and is necessary to operate the Guard:

- Dragonfly Guard Power Supply, Manufactured by Phihong, Part No 552-PSA-30u-050;
- Dragonfly Guard Power Cord, Manufactured by Belden, Part No Bel-17251-B1-10.
- Dragonfly Ignition Card, Advanced Micro Devices C-Series Flash Memory Card, 2Mbytes, AmC002CFLKA.
- Mykotronx Fortezza® Crypto Card or equivalent.

8.1.2 Evaluated Software Components

The software stored on the internal flash memory card configured as Floppy Drive A: of the Guard consists of the following:

- DOS Version 6.2
- autoexec.bat (dated 2/4/98);
- config.sys (dated 6/12/96);
- dpcmcia.exe (dated 11/3/97);
- verifile.exe (dated 2/4/98);
- chk4igni.exe (dated 10/8/97);
- startup.bat (11/26/97);
- trouble.exe (dated 10/21/97).

The PIN for the Fortezza Card and the following files are stored on the Ignition Card:

- dlfy.exe (dated 980821.1539; this is software release 3.0)
- dpcmcia.exe (dated 11/3/97)
- startup.bat (dated 11/3/97)
- setport.exe
- fortload.exe
- slip8250.exe (Not used in Evaluated Configuration)
- hdlc8250.com (Not used in Evaluated Configuration)
- ne2000.com
- set4i29.exe
- dpmi16bi.ovl (8/29/95)
- rtm.exe (8/29/95)

The Fortezza Card contains its PIN and the user's private key as well as the following certificates:

- Root Authority Certificate (root here is ITT)
- Root Certificate, signed by the Root Authority
- Local Authority Certificate, signed by the Root
- User Certificate, signed by the Local Authority
- Audit Mask Certificate, signed by the Local Authority
- Certificate Revocation List, signed by the Local Authority
- Configuration Certificate, signed by the Local Authority

- Routing Certificate, signed by the Local Authority.

8.2 Configuration and Usage Notes

The Configuration Certificate on the Fortezza Card contains configuration options for the evaluated configuration. The Guard cannot change these options; instead, the Fortezza Card must be updated by the Administration System under control of the Local Authority. The following sections cover required and allowed configuration settings, configuration settings that are not allowed in the evaluated configuration, and incorrect installation of the evaluated configuration.

8.2.1 Required and Allowed Configuration Settings

The Administration System forces a selection in the case of some required fields. For example, security level must be one of Unclassified, Sensitive but Unclassified, Confidential, Secret, or Top Secret. Such configuration settings are not listed here. Configuration settings with standard defaults (e.g., "Max Crypto Period" for key expiration, or "Association time to live" for how long an association can exist without activity) are not listed either. Other configuration settings can be left blank or “none” can be selected, allowing a security functional requirement to be unsatisfied. These configuration options must be set as noted to support the evaluated configuration:

Configuration Option	**Required Setting**
Requires Audit Catcher	This must be set to “yes” unless the Dragonfly Guard is itself an Audit Catcher, in order to guarantee that the Guard’s CRL and Audit Mask are updated from the Audit Catcher. See DF_ST security functional requirement FAU_GEN.1.1, FAU_SEL.1.1.
Audit Mask	This must be set to “standard” or “audit all” if auditing is desired or to demonstrate the TOE is able to generate audit events. If set to “standard”, then all audit events will be audited initially but updates to the audit mask will be made from the Audit Catcher when the Audit Catcher’s audit mask is updated. If the Guard is itself an Audit Catcher, then the mask must be set to “standard” to allow updates of the reporting Guard’s CRL and audit mask. If set to “audit all”, all audit events will be audited. See DF_ST security functional requirement FAU_GEN.1.1, FAU_SEL.1.1.

Table 8-1. Required Configuration Options

The following configuration options may be set as part of the evaluated configuration:

Configuration Option	**Comments**

Local Audit Catcher	Audit catcher can be reached from local port.
Remote Audit Catcher	Audit catcher can be reached from remote port.
Is Audit Catcher	“yes” if it is, “no” otherwise.
Local Privilege Vector	To enforce DAC during associations established through the local port.
Remote Privilege Vector	To enforce DAC during associations established through the remote port.
Local Firewall	To enable Firewall Mode for packets coming in through the local port.
Remote Firewall	To enable Firewall Mode for packets coming in through the remote port.
Allow Write Ups	To allow write ups from the low side to the high side of the Guard.
Proxy RARP	To enable RARP requests and responses to pass through the Guard.
Proxy ARP	To selectively allow or prohibit ARPs to selected Domains.

Table 8-2. Optional Configuration Settings

ARP and RARP are the only configurable services.

8.2.2 Non-Evaluated Configuration Settings

The following table summarizes configuration options supported by the Dragonfly Guard that are NOT part of the evaluated configuration.

Configuration Option	**Comments**
Point to Point Protocol (PPP) and Serial Line Interface Protocol (SLIP)	The serial port is only used for audit output in the evaluated configuration.
Local port ARP accept table, Local port ARP deny table	These are used for Guards that interface with a PPP or a SLIP line.
Remote port ARP accept table, Remote port ARP deny table	These are used for Guards that interface with a PPP or a SLIP line.
TNS/IGW option	The Tactical Packet Network is not included in the evaluated configuration.

Table 8-3. Non-Evaluated Configuration Settings

8.2.3 Incorrect Installation of the Evaluated Configuration

The most serious problem with installation of the Dragonfly Guard occurs if the Local and Remote ports are physically connected to the wrong networks. The Dragonfly Guard can do no checking for this, since it depends on the User Fortezza Card’s Configuration Certificate for port security labels. The DF_GUM contains a warning about reversing the Local and Remote network connections. The Guard administrator should expect a problem if a Guard is not checking in with its Audit Catcher.

Another installation problem can result from incorrect preparation of the Configuration Certificate on the User Fortezza Card. The Administration System can do limited checking for this when the configuration options are selected. For example, if write-ups are enabled, but the Local port is the same security level compared to the Remote Port, then a configuration error may have occurred. However, the Dragonfly Guard itself cannot change anything on the User Fortezza Card.

DF_GUM section 4.1.1 gives a sequence of general corrective measures to follow in case of problems. Installation problems that result from initialization failures, self-test errors, bad connections, operational problems such as expiration of a certificate, etc. are indicated by light flash sequences documented in section 4.1.3 of the DF_GUM. Section 4.1.2 gives a sequence of steps to follow to check the Guard's configuration data during operation of the Guard.

8.3 Target Environment

The Dragonfly Guard is intended for environments where legacy IP-based networks connect domains of hosts operating at different security levels. The Dragonfly Guard uses a Fortezza Card to provide MLS services over such IP-based networks.

Several configurations of the Dragonfly Guard have been described in section 4. Of those described, the Single Dragonfly Guard between two Domains, the Dragonfly Guard for each Domain, and the Mixed Enclave configurations are included in the evaluated configuration as long as all connections shown do not involve PPP, SLIP, or other dial-up protocols. Instead, all connections shown are direct wire connections.

The Designated Dragonfly configuration used with a firewall was not included in the configuration that the evaluation team examined, and therefore is not in the evaluated configuration.

8.4 Residual Vulnerabilities

Assuming that the Dragonfly Guard is installed, configured, and operated correctly, the following vulnerabilities remain. However, these vulnerabilities may be countered by procedural or physical means.

- **Malicious or careless administrator at Local Authority.** The Dragonfly Guard depends on the Fortezza Card inserted into it for its configuration information, for identification and authentication, and for cryptographic services. If the Fortezza Card is not properly prepared by use of the Administration System, the Dragonfly Guard will not operate as expected.

- **Incorrect or Compromised Administration System.** The Administration System is outside of the evaluated configuration, but the Dragonfly Guard depends on the correct operation of the Administration System to produce the Fortezza Card and to modify the security attributes that are on the Fortezza Card.

- **Overrun.** If the Dragonfly Guard unit, with its Ignition Card and Fortezza Card, are captured by an adversary, they will continue to function until the certificates on the Fortezza Card are revoked.

9 Results of Evaluation

The Dragonfly Guard was found to meet all the functional requirements from the Security Target and all the assurance requirements of Evaluation Assurance Level 2, as specified by the Security Target. Section 9.1 states each functional requirement and then explains how the Guard meets the requirement, including the functions from the Security Target that support meeting the requirement. Section 9.2 describes how the Guard meets the Strength of Function claim of Medium. Section 9.3 consists of a table that verifies the Guard meets the requirement in the first column, names the requirement in the second column, and states the requirement in the third column together with an explanation of how the Guard meets the requirement.

9.1 TOE Security Functional Requirements

1. FAU_GEN.1 Audit data generation

Hierarchical to: No other components.

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;
b) All auditable events for the [*not specified*] level of audit; and
c) [*Closing a Write Up,*
d) *Anticipated Message Mismatch,*
e) *Anticipated Message Not allowed,*
f) *Anticipated Message Unknown,*
g) *Association Request Denied (Reported by Responder),*
h) *Association Request Denied (Reported by Initiator),*
i) *Association Closed,*
j) *Association Granted,*
k) *Association Requested,*
l) *Association Unknown,*
m) *Audit Mask Received,*
n) *Opening a Write Up Session,*
o) *Certificate or Symmetric Key Deleted,*
p) *Invalid Signature,*
q) *Lost Wait Queue Msg,*
r) *Received by non-Audit Catcher,*
s) *Certificate Revocation List Sent,*
t) *Old CRL Version,*
u) *Certificate Invalid Start,*
v) *Certification Expired,*
w) *Certificate Revoked,*
x) *Certificate Invalid, and*
y) *Security Level Mismatch.]*

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information: Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event.
Note: If a Guard is configured to generate audit event messages, it will never generate an audit record of the shutdown of the audit functions, because the Guard never stops auditing after it starts and is still operating.
Dependencies: FPT_STM.1 Reliable time stamps

Every Guard can be configured by the Administration System to generate audit events. These events include all of the ones in the requirement (c-y), as specified by function AUDIT-6 in the Security Target. The audit events are sent to the Guard's designated Audit Catcher (a Guard, either the same or different), which is configured by the Administration System. Startup of audit functions is also shown in the audit log by the first check-in message from a Guard sent to its audit catcher and by the first local status message sent by the audit catcher to its audit log. An audit record for the shutdown of audit functions is never generated because, if the Guard is configured to generate audit messages, auditing cannot be shutdown while the Guard is operational. The audit catcher sends the audit log to its serial port for display. Audit records include the Guard Unit and its IP address (these constitute the subject identity), the audit code and brief description (these constitute the type of event and the outcome), and the date and time. Each event code can be can be interpreted as a success or failure depending on its meaning as described in the User Manual. The requirement is met by functions AUDIT-1:6 in the Security Target.

2. FAU_SEL.1 Selective audit
Hierarchical to: No other components.
FAU_SEL.1.1 The TSF shall be able to include or exclude auditable events from the set of audited events based on the following attribute: [*event type*].

Dependencies: FAU_GEN.1 Audit data generation
FMT_MTD.1 Management of TSF data
ITENV.3 Dragonfly Administration System for Setting User Attributes
ITENV.4 Dragonfly Administration System for Modifying TSF Data

The events audited can be set by the Administration System. Events are selected by setting bits in a 256-bit vector (the Audit Mask field) in the Audit Mask certificate, which is stored on the User Fortezza card. The Guard will then include or exclude events based on the settings of the Audit Mask on the User Fortezza card that is required for operation of the Guard. In addition, if the Standard Audit Mask is set on both the Guard and its Audit Catcher, the selection of events to be audited can be changed by the Audit Mask of the Audit Catcher. The Audit Catcher's Audit Mask will be distributed to the Guards that designate it as their Audit Catcher. The requirement is met by AUDIT-6:8 and SM-3 in the Security Target.

3. FDP_ACC.1 Subset access control
Hierarchical to: No other components.
FDP_ACC.1.1 The TSF shall enforce the [*discretionary access control SFP*] on [
a) subject: source domain,

b) object: destination domain, and

c) *operation: release to.*]

Dependencies: FDP_ACF.1 Security attribute based access control

One or more hosts connected to a Guard Ethernet port are considered a Dragonfly Domain. A Dragonfly Domain is a set of hosts, without any other intervening Guard, at the security level of the Guard port they are connected to. Even if MAC policy does not prevent communication between Domains, Privilege Vectors can be used to prevent communication. There is a Privilege Vector for each port of a Guard. Each bit in a Privilege Vector represents a known Domain. If the bit is set in the Privilege Vector at either the source or destination Guard, communication between Domains is allowed (subject to MAC policy); otherwise, it is prevented. The Privilege Vector is checked when an association is requested. A second form of DAC is enforced by configuring a port to be in Firewall Mode. All native packets can be prevented from being accepted from or released to any non-Dragonfly protected hosts connected to a port configured in Firewall Mode. If a Guard port is configured in Firewall Mode, no native packets will be released to the destination Domain connected to the port, and no native packets (that is packets not from a Guard) will be accepted from a source Domain connected to the port. DAC-1, DAC-2, and IP-2 are the functions described in the Security Target that meet this requirement.

4. FDP_ACF.1 Security attribute based access control

Hierarchical to: No other components.

FDP_ACF.1.1 The TSF shall enforce the [*discretionary access control SFP*] to objects based on [privilege vectors or firewall mode].

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [

1) If there are two or more Dragonfly Guards between the source domain and the destination domain, then

a) [If the source domain privilege vector has the bit set for the destination domain, then the datagram is released if the MAC check passes, or

b) If the destination domain privilege vector has the bit set for the source domain, then the datagram is released if the MAC check passes,

c) Else the datagram is not released.]

2) If there is only one Dragonfly Guard between the source domain and the destination domain and firewall mode is disabled (i.e., native mode communication is allowed), datagrams are released if they pass the MAC checks.

3) If the Dragonfly Guard has Firewall Mode enabled for a port, no datagrams may be received from or released to a Native host in the domain associated with that port].

Note: A Dragonfly Guard with Firewall Mode enabled for a port will not be able to communicate with hosts attached directly to that port.

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [*none].*

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the [*no additional rules].*

Dependencies: FDP_ACC.1 Subset access control
FMT_MSA.3 Static attribute initialisation

ITENV.3 Dragonfly Administration System for Setting User Attributes

MAC is always enforced by the use of security labels placed on native packets and subsequent checking before release. Even if MAC policy enforcement allows communication between hosts, a separate DAC policy can be enforced through the use of privilege vectors or Firewall Mode for one of a Guard's ports. Privilege vectors apply to each port of a Guard and can prevent communication between Domains of hosts (not individual hosts within a Domain) independent of their security levels. Privilege vectors for each port are stored in each Guard's User certificate and these are exchanged during the association process. The Guard granting the association checks both certificates' appropriate privilege vector (for the local or remote port depending on what is source and destination Domain). If neither Guard's privilege vector contains the appropriate Domain (one containing the source or destination) then the association is denied. Even if MAC and Privilege Vector configuration would allow communication between hosts, if one of the hosts is part of a Domain whose Guard port is configured in Firewall Mode, no communication can occur with that host. DAC-1, DAC-2, and IP-2 are the functions that meet this requirement.

5. FDP_ETC.1 Export of user data without security attributes

Hierarchical to: No other components.

FDP_ETC.1.1 The TSF shall enforce the [*mandatory access control SFP*] when exporting user data, controlled under the SFP(s), outside of the TSC.

FDP_ETC.1.2 The TSF shall export the user data without the user data's associated security attributes.

Dependencies: [FDP_ACC.1 Subset access control, or
FDP_IFC.1 Subset information flow control]
ITENV.3 Dragonfly Administration System for Setting User Attributes

Note: FDP_ETC.1 applies only when data is exported to a native host. In this case, the host is in the same security Domain and has the same security attributes as the port from which the data is exported.

All hosts that are connected to a Dragonfly Ethernet port without any other Dragonfly between them and the port are assumed to be at the single security level of the port. No data is released to these hosts in violation of mandatory access control policy. If there are no MAC violations (and Firewall Mode is not enabled), only unencrypted user data are released to these hosts; the security level and checksum are never released. The security level and checksum are security attributes. This requirement is met by the functions, EXP-1 and SL-3.

6. FDP_IFC.1 Subset information flow control – Mandatory Access Control SFP

Hierarchical to: No other components.

FDP_IFC.1.1 The TSF shall enforce the [*mandatory access control SFP*] on [

a) Subjects: Dragonfly domains,

b) Information: IP datagrams,

c) *Operation: release from source domain to destination domain*.]

Dependencies: FDP_IFF.1 Simple security attributes

MAC policy checks are made either once or twice depending on whether the destination Domain is protected by a different Guard than the source Domain. If there are at least two Guards involved, a check is made at the Guard connected to the source Domain before an IP datagram is encrypted and sent to a destination Domain protected by another Guard. The other check is always made before a native IP datagram is released to a destination Domain. The details are described in section 3.2, Mandatory Access Control Policy, and section 5.3.4, Dragonfly Operation. This requirement is met by functions, MAC-1:10.

7. FDP_IFF.2 Hierarchical security attributes – Mandatory Access Control SFP

Hierarchical to: FDP_IFF.1

FDP_IFF.2.1 The TSF shall enforce the[*mandatory access control SFP*] based on the following types of subject and information security attributes: [

a) Security level of the source domain,

b) Security level of the destination domain,

c) Type of protocol (ARP, RARP, ICMP, UDP, TCP, FTP, and SMTP, and DNS),

d) *Type of request, response or command,*

e) Writeups enabled,

f) ARP Proxy is allowed, and

g) *RARP Proxy is allowed*.]

FDP_IFF.2.2 The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules, based on the ordering relationships between security attributes hold: [

A) If the security levels of the source domain and destination domain are equal, release the IP datagram.

B) If the security level of the destination domain is greater than the security level of the source domain (writeup), the following rules apply based on the type of protocol:

1) Address Resolution Protocol (ARP)/Reverse Address Resolution Protocol (RARP): If ARP proxy is allowed, ARP Requests and Responses are allowed. If the RARP proxy is allowed, RARP Requests and Responses are allowed.

2) If writeups are enabled, the following rules apply:

a) Internet Control Message Protocol (ICMP): Echo Requests and Time Stamp Requests are allowed.

b) User Datagram Protocol (UDP): Domain Name Server Requests with the one question flag set are allowed.

c) Transmission Control Protocol (TCP): Domain Name Server Requests with the one question flag set are allowed.

d) File Transfer Protocol (FTP): The following FTP commands are allowed: ABOR, ACCT, ALLO, APPE, CWD, MODE, NOOP, PASS, PORT, PWD, QUIT, STOR, STOU, STRU, TYPE, USER, and XPWD.

e) *Simple Mail Transfer Protocol (SMTP): The following SMTP Commands are not allowed: EXPN, HELP, LIST, RETR, STAT, TOP, and TURN. Everything else is allowed.*

f) All other messages types are released.

Note: However, since predicted responses are not generated for these message types, any replies to them will be blocked.

C) If the security level of the destination domain is less than the security level of the source domain (writedown), only ARP/RARP requests/responses and predicted messages are released as described below:
When the Dragonfly Unit allows a write up to occur, i.e., releases an ARP/RARP request or an IP datagram to a destination domain at a higher security level, the Dragonfly Guard shall generate a predicted response at the level of the source domain. When the Dragonfly Guard receives an actual response from the destination domain, it shall compare the actual response with the predicted response. If the actual response matches the predicted response, the Dragonfly Unit, shall copy only the fields containing control information (i.e., not user data) specified in the High Level Design from the actual response to the predicted response.

Predicted Responses are listed below by type of protocol. Predicted responses are only released if the actual response matches the predicted response.

1) Address Resolution Protocol (ARP) /Reverse Address Resolution Protocol (RARP): If the ARP proxy is allowed, ARP requests and responses are allowed; If the RARP proxy is allowed, RARP requests and responses are allowed.

2) If writeups are enabled, the following rules apply:

a) Internet Control Message Protocol (ICMP): The following responses are allowed, ICMP Echo Responses, ICMP Time Stamp Responses, ICMP Unreachable Destination, ICMP Source Quench, and ICMP Time Exceeded.

b) User Datagram Protocol (UDP): Domain server responses with only one answer are allowed.

c) Transmission Control Protocol (TCP): Domain server responses with only one answer are allowed.

d) File Transfer Protocol (FTP): Predicted responses to the allowed commands that match the actual responses are allowed.

e) *Simple Mail Transfer Protocol (SMTP): Predicted responses to the allowed commands that match the actual responses are allowed.]*

FDP_IFF.2.3 The TSF shall enforce [*no additional mandatory access control SFP rules*].
FDP_IFF.2.4 The TSF shall provide [*no additional mandatory access control SFP capabilities*].
FDP_IFF.2.5 The TSF shall explicitly authorise an information flow based on the following rules: [*no additional rules]*.
FDP_IFF.2.6 The TSF shall explicitly deny an information flow based on the following rules: [*no additional rules]*.
FDP_IFF.2.7 The TSF shall enforce the following relationships for any two valid information flow control security attributes:

a) There exists an ordering function that, given two valid security attributes, determines if the security attributes are equal, if one security attribute is greater than the other, or if the security attributes are incomparable; and

b) There exists a “least upper bound” in the set of security attributes, such that, given any two valid security attributes, there is a valid security attribute that is greater than or equal to the two valid security attributes; and

c) There exists a “greatest lower bound” in the set of security attributes, such that, given any two valid security attributes, there is a valid security attribute that is not greater than the two valid security attributes.

Note: The TSF supports the following set of hierarchical security levels: Unclassified, Sensitive But Unclassified (SBU), Confidential, Secret and Top Secret.
Dependencies: FDP_IFC.1 Subset information flow control
FMT_MSA.3 Static attribute initialisation
ITENV.3 Dragonfly Administration System for Setting User Attributes

The details about how this complicated requirement is met are described in section 3.2, Mandatory Access Control Policy, and section 5.3.4, Dragonfly Operation. ARP and RARP packets are not IP datagrams, and no MAC checks are made for these. ARP and RARP are proxied depending on the Proxy RARP setting and the ARP Accept and Deny tables configuration. These settings are not relevant to the evaluated configuration, which does not allow use of the serial port for PPP or SLIP. MAC policy is checked for IP datagrams. Logically, this is what happens: the security labels are checked. If they are equal, all IP datagrams are released. If they are unequal, the Guard process checks if write-up is enabled. If write-up is not enabled, the datagram is discarded; if it is enabled, it is processed in an application protocol dependent way, as explained in section 3.2. For each protocol (ICMP, SMTP, FTP, and DNS), requests and responses are handled differently depending on if it is a write-up (allowed by MAC policy in this case) or write-down (either disallowed or constrained to pass minimal control information to make the protocol work). A disallowed write-down (that is, a response to an allowed write-up) occurs when there is no anticipated message and can generate an audit message, "Anticipated Message Unknown." This requirement is met by functions, SL-1:3, MAC-1:10, and IP-6.

8. FDP_ITC.1 Import of user data without security attributes
Hierarchical to: No other components.
FDP_ITC.1.1 The TSF shall enforce the [*mandatory access control SFP*] when importing user data, controlled under the SFP, from outside of the TSC.
FDP_ITC.1.2 The TSF shall ignore any security attributes associated with the user data when imported from outside the TSC.
FDP_ITC.1.3 The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: [*None]*
Dependencies: [FDP_ACC.1 Subset access control, or
FDP_IFC.1 Subset information flow control]
FMT_MSA.3 Static attribute initialisation
ITENV.3 Dragonfly Administration System for Setting User Attributes
Note: FDP_ITC.1 applies only when data is imported from a native host. In this case, the host is in the same security domain and has the same security attributes as the port on which the data is imported.

A native packet (a network packet generated by a host and thus not encapsulated as a result of Dragonfly processing) is not labeled with a security level or any other security attribute. Any indication of a security level in the user data (such as "Subject: Top Secret Information") would be ignored by the Guard. A native packet is received by a Guard on one of its two Ethernet ports. Each port is configured at a single security level; the two ports may be at the same level or different levels. When a native packet is received by a Guard (and Firewall Mode is not enabled), the destination address is checked in the host table, which points to the association table, which stores

the security level of Guards with which associations have been previously established. If the destination is not found, the process of association establishment begins. Otherwise, if the security level of the input port is equal to the security level of the destination, or write up is allowed, the packet is processed and eventually a security level equal to the input port is appended to the packet along with a checksum. Thus, all such packets are labeled with the security level of the Dragonfly port on which the packet was received. This is how the two functions, IMP-1 and SL-3, meet the security requirement.

9. FDP_UCT.1 Basic data exchange confidentiality

Hierarchical to: No other components.

FDP_UCT.1.1 The TSF shall enforce the [*mandatory access control SFP*] to be able to [*transmit and receive*] objects in a manner protected from unauthorised disclosure.

Dependencies: [FTP_ITC.1 Inter-TSF trusted channel, or
FTP_TRP.1 Trusted path]
[FDP_ACC.1 Subset access control, or
FDP_IFC.1 Subset information flow control]
ITENV.1 Cryptographic Services on the Fortezza Card

Note: Although data confidentiality supports MAC, data confidentiality is provided independently of the mandatory access control SFP.

The Guard uses its Fortezza card to encrypt and decrypt all network packets to and from another Guard. A symmetric key is generated during association establishment between two Guards and used to encrypt and decrypt. Network packets are only released in plaintext form only after MAC policy has been checked (assuming Firewall Mode is not enabled). This requirement is met by functions, ASSOC-3, ASSOC-4, IP-1, IP-5, and CONF-1.

10. FDP_UIT.1 Data exchange integrity

Hierarchical to: No other components.

FDP_UIT.1.1 The TSF shall enforce the [*mandatory access control SFP*] to be able to [*transmit and receive*] user data in a manner protected from [*modification, deletion, or insertion*] errors.

FDP_UIT.1.2 The TSF shall be able to determine on receipt of user data, whether [*modification, deletion, or insertion]* has occurred.

Note: Although data integrity supports MAC, data integrity is provided independently of the mandatory access control SFP.

Dependencies: [FDP_ACC.1 Subset access control, or
FDP_IFC.1 Subset information flow control]
[FTP_ITC.1 Inter-TSF trusted channel, or
FTP_TRP.1 Trusted path]
FDP_UIT.1 Cryptographic Services on the Fortezza Card

User data, sent from one Guard to another Guard, is protected from modification, deletion, or insertion by calculating a 32-bit checksum to datagrams containing user data. The user data, security label, and checksum are encrypted using the Fortezza card service before transmission. Upon reception, the datagram is decrypted, and the checksum is recalculated and compared with the decrypted checksum. If the checksum calculated after decryption is different from the checksum

calculated before encryption, the datagram is discarded, and the detection of an integrity violation can be audited. (An integrity violation would generate the "Invalid Signature" audit event.) This requirement is met by functions, IP-1, IP-5, and INT-1.

11. FIA_ATD.1 User attribute definition

Hierarchical to: No other components.

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: *[*

a) User Certificate,
b) Configuration Certificate,
c) Audit Certificate,
d) Certificate Revocation List certificate, and
e) Cryptographic Keys]

Note: The user is the Dragonfly Guard itself. The user attributes contained in the User Certificate, Configuration Certificate, Audit Certificate, and Certificate Revocation List certificate are stored on the User Fortezza Card. These attributes are set by the Dragonfly Administration System. Cryptographic keys are generated by the cryptographic services on the User Fortezza Card during TOE operation.

Dependencies: ITENV.1 Cryptographic Services on Fortezza Card
ITENV.3 Dragonfly Administration System for Setting User Attributes

The certificates that are security attributes are configured by the Dragonfly Administration System and stored on the User Fortezza card required for the operation of each Guard, the only "user" with respect to this requirement. The Routing certificate could also be used to store user attributes but is not used for the evaluated configuration. The certificates cannot be changed during operation of the Guard in the evaluated configuration. Possibly multiple cryptographic keys are stored by the Guard in its symmetric key list pointed to by its association list entries for each association established with other Guards. This requirement is met by the functions, ATTR-1 and SM-2.

12. FIA_UAU.2 User authentication before any action

Hierarchical to: FIA_UAU.1

FIA_UAU.2.1 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Dependencies: FIA_UID.1 Timing of identification
ITENV.1 Cryptographic Services on the Fortezza Card

The Guard requires a User Fortezza card to operate. The Dragonfly software logs on to the User Fortezza card with the Fortezza card PIN. The PIN is stored in Dragonfly software on the ignition card. All this happens before any network packet processing, including association establishment, is performed. Network packet processing are the TSF-mediated actions performed on behalf of the "user," the Guard itself. One Guard is authenticated to another Guard during the association process, when User certificates are exchanged. See steps 7 and 17 in section 5.3.4. This requirement is met by the functions, ASSOC-2 and IA-1:3.

13. FIA_UID.2 User identification before any action

Hierarchical to: FIA_UID.1
FIA_UID.2.1 The TSF shall require each user to identify itself before allowing any other TSF-mediated actions on behalf of that user.
Dependencies: No dependencies.

Each Guard is uniquely identified by its User certificate stored on the User Fortezza card. The Guard cannot boot up or operate without a User Fortezza card and User certificate. This User certificate is used in association establishment with other Guards to identify the Guard. No TSF-mediated actions occur before identification is complete (User certificate is validated and its fields used for configuring the Guard). This requirement is met by the functions, IA-1:3.

14. FMT_MTD.1 Management of TSF data
Hierarchical to: No other components.
FMT_MTD.1.1 The TSF shall restrict the ability to [set] the [audit mask and certificate revocation list] to [the local authority].
Dependencies: FMT_SMR.1 Security roles
ITENV.4 Dragonfly Administration System for Modifying TSF Data

The audit mask and the Certificate Revocation List can only be set on the User Fortezza card of a Guard using the Administration System, which requires the Local Authority Fortezza card. If the Guard is an audit catcher for another Guard with its audit mask set to "standard", then the other Guard's audit mask and Certificate Revocation List will be updated after the audit catcher Guard is reinitialized and the Guard(s) designating it as their audit catcher check-in. In any case, the setting of the audit mask and Certificate Revocation List is ultimately under the control of the "Local Authority". This requirement is met by the function, SM-3.

15. FMT_REV.1 Revocation
Hierarchical to: No other components.
FMT_REV.1.1 The TSF shall restrict the ability to revoke security attributes associated with [*a Dragonfly Guard*] within the TSC to [*the local authority*].
FMT_REV.1.2 The TSF shall enforce the rules: [*If a certificate appears on a Dragonfly Guard's Certificate Revocation List, the Dragonfly Guard will reject packets originating from a Dragonfly Guard using that Certificate].*
Note: The TSF provides the ability to revoke certificates which contain security attributes.
Dependencies: FMT_SMR.1 Security roles
ITENV.3 Dragonfly Administration System for Setting User Attributes
ITENV.4 Dragonfly Administration System for Modifying TSF Data

The Certificate Revocation List (CRL) is stored on the Certificate Revocation List certificate and can only be set on the User Fortezza card of a Guard using the Administration System (although it can be updated by an Audit Catcher's more recent CRL). The Administration System requires the Local Authority Fortezza card. A Guard is identified by its User Certificate. If its User Certificate is on the Certificate Revocation List, it cannot establish associations with other Guards. The CRL is checked when the destination Domain's Guard receives an association request. If the User

Certificate of the Guard requesting the association is on the CRL, an association denied message will be sent back to it. This requirement is met by the functions, CRL-1, CRL-2, and SM-3.

16. FMT_SAE.1 Time-limited authorisation

Hierarchical to: No other components.

FMT_SAE.1.1 The TSF shall restrict the capability to specify an expiration time for [*User Certificates and cryptographic keys*] to [*the local authority*].

FMT_SAE.1.2 For each of these security attributes, the TSF shall be able to *[not accept packets originating from a Dragonfly Guard using a User Certificate*] after the expiration time for the [certificate or cryptographic key*]* has passed.

Dependencies: FMT_SMR.1 Security roles
FPT_STM.1 Reliable time stamps
ITENV.3 Dragonfly Administration System for Setting User Attributes

The User certificate stores a start time and an expiration period from the start time that defines when the certificates on the User Fortezza card are valid. (The CRL and the Audit Mask certificates also store an expiration time, but a sequentially generated certificate "ID NUMBER" in those certificates is used as a revision number to check their validity.) The Configuration certificate stores the maximum length of time an association can exist as well as a maximum period of inactivity allowed for an association (and for which periods the symmetric key for the association is valid). These parameters can only be set using the Administration System, which requires the Local Authority Fortezza card. The expiration time for the User certificate is checked during association establishment by both the requesting and potentially granting Guards. If the expiration time has passed, the association cannot be established. The inactivity expiration period for the association key is updated on decryption. Periodically (independent of datagrams needing processing) the maximum length of time for an association is checked. If it has been exceeded, the entry for the association is deleted, and a new association will be required for communication between those Domains. This requirement is met by the functions, ATTR-2 and ATTR-3.

17. FMT_SMR.1 Security roles

Hierarchical to: No other components.

FMT_SMR.1.1 The TSF shall maintain the roles [*User*].

FMT_SMR.1.2 The TSF shall be able to associate users with roles.

Note: Certificates for the root authority, root, local authority, and user are stored on the User Fortezza Card for the Dragonfly Guard, but the Dragonfly Guard only assumes the role of User. The TSF associates the Dragonfly Guard user with the User Role when the Dragonfly Guard software logs into the User Fortezza Card using the PIN for the User Certificate.

Dependencies: FIA_UID.1 Timing of identification
ITENV.5 Certificates on the Fortezza Card
ITENV.6 Fortezza Card PINs

Each Guard maintains the User role in the User certificate on the User Fortezza card that must be inserted in one of the Guard's PCMCIA slots for Guard operation. The operational Guard cannot change the User certificate corresponding to the User role; hence, the role is maintained by the TSF as long as the User Fortezza card is inserted. The Guard's software must be able to login to the User

Fortezza card using a PIN stored in the Guard's software. The Guard is thus associated with the User role represented by the User certificate. It is the User certificate that identifies a Guard to other Guards and is exchanged during association establishment. This requirement is met by functions, IA-1, IA-2, and SM-1.

18. FPT_AMT.1 Abstract machine testing

Hierarchical to: No other components.
FPT_AMT.1.1 The TSF shall run a suite of tests [*during initial start-up*] to demonstrate the correct operation of the security assumptions provided by the abstract machine that underlies the TSF.
Dependencies: No dependencies.

The abstract machine that underlies the TSF is a commodity 486 motherboard running MS-DOS 6.2. Its correct operation is demonstrated by the successful completion of the Power On Self Tests of the CPU and the BIOS checks of the CPU peripherals and memory. (The Fortezza card also runs a suite of self tests.) If the tests succeed the Guard will emit a buzz 15-20 seconds after power is turned on and then a single beep several seconds later. If any of the tests fail, the Guard will not operate and there will be an error code indicated by the red fault light staying on and the green ready light flashing two sequences that represent the first and second digit of the error code, which can be looked up in the user manual. This requirement is met by function, INIT-1.

19. FPT_ITI.1 Inter-TSF detection of modification

Hierarchical to: No other components.
FPT_ITI.1.1 The TSF shall provide the capability to detect modification of all TSF data during transmission between the TSF and a remote trusted IT product within the following metric: [*based on the cryptographic services provided by the User Fortezza Card.]*
FPT_ITI.1.2 The TSF shall provide the capability to verify the integrity of all TSF data transmitted between the TSF and a remote trusted IT product and [*reject the IP datagram*] if modifications are detected.
Note: IP Datagrams containing TSF Data are either hashed and digitally signed or a checksum is computed and the message and checksum are encrypted using a symmetric key.
Dependencies: ITENV.1 Cryptographic Services on Fortezza Card

Association requests, grants, denials, and association unknown messages are signed datagrams using the Fortezza card service for digital signature with the User Certificate's private key. Audit-related messages and CRL messages are Protected Dragonfly Messages, which have a checksum appended to the TSF data and then encrypted using Fortezza for modification detection. If either the digital signature or the checksum is invalid, the datagram is discarded and the event can be audited (generating an "Invalid Signature" audit message). This requirement is met by the functions, ASSOC-2, ASSOC-3, IP-1, IP-4, IP-5, and INT-2.

20. FPT_RVM.1 Non-bypassability of the TSP

Hierarchical to: No other components.
FPT_RVM.1.1 The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.
Dependencies: No dependencies.

Packets are native user packets (IP datagrams or ARP/RARP Ethernet frames), protected user datagrams, signed datagrams, or protected Dragonfly datagrams. Each of these types are checked according to the security policy regarding MAC, DAC, Firewall Mode, write-up enabled, and ARP/RARP processing before they are released by the Guard. This requirement is met by the function, SA-1.

21. FPT_SEP.1 TSF domain separation

Hierarchical to: No other components.
FPT_SEP.1.1 The TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.
FPT_SEP.1.2 The TSF shall enforce separation between the security domains of subjects in the TSC.
Note: There is only one security domain on the Dragonfly Guard, the one that the Dragonfly Guard executes its own code in. No other code is executed on a Dragonfly Guard.
Dependencies: No dependencies.

There is only one domain of execution on the Guard. It executes the code on the flash floppy drive inside the Dragonfly case, which cannot be removed while the Guard is physically protected, and the digitally signed Dragonfly code loaded from the ignition card. Code that could be a threat to its execution cannot be loaded through either the serial or network interfaces. This requirement is met by the function, SA-2.

22. FPT_STM.1 Reliable time stamps

Hierarchical to: No other components.
FPT_STM.1.1 The TSF shall be able to provide reliable time stamps for its own use.
Dependencies: No dependencies.

The Fortezza card's clock is initialized during its configuration by the Administration System. After the Dragonfly software is loaded, it logons to the Fortezza card, reads the Fortezza card's internal clock, and sets the motherboard's system clock. The system clock then provides reliable time stamps for the Guard's own use. This requirement is met by the function, TIME-1.

23. FPT_TDC.1 Inter-TSF basic TSF data consistency

Hierarchical to: No other components.
FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret [*all security attributes*] when shared between the TSF and another trusted IT product.
FPT_TDC.1.2 The TSF shall use [*the following rule: the security attributes received from another TOE's TSF (i.e., another Dragonfly Guard) mean the same on the TSF at which it is received*] when interpreting the TSF data from another trusted IT product.
Dependencies: No dependencies.
Note: Dragonfly Guards only interpret TSF data from other Dragonfly Units.

Dragonfly Guards only interoperate with other Dragonfly Guards in terms of shared security attributes. (Native hosts connected to a Dragonfly port do not have security aspects.) Hence, the interpretation of security levels, privilege vectors, and other security attributes is consistently

interpreted between Guards; for example, a Secret level means the same at one Guard as another. This requirement is met by the function, CONS-1.

24. FTP_ITC.1 Inter-TSF trusted channel

Hierarchical to: No other components.

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and a remote trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure. .

Note: Dragonfly messages containing TSF Data that needs to be protected from disclosure are encrypted. Dragonfly Messages that require protection from modification but not disclosure such as Association Request and Grant messages are digitally signed, but not encrypted.

FTP_ITC.1.2 The TSF shall permit [either *the TSF or the remote trusted IT product*] to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for [*communication with another Dragonfly Unit]*.

Dependencies: ITENV.1 Cryptographic Services on Fortezza Card

Two Guards use Fortezza services to establish a logically distinct channel between them, known as an association. Endpoints are uniquely and unequivocally identified during association establishment with a signed User certificate representing each endpoint. Each channel between Guards has its own unique key for encryption. All data flowing through such a channel is protected from disclosure by Fortezza encryption (Skipjack) and protected from modification by a 32-bit checksum. This requirement is met by the functions, ASSOC-1:4.

9.2 Strength of Function (SOF) Requirement

The Security Target claims that the minimum strength of function level for the TOE security functional requirements is SOF-medium. No claim about specific metrics is made for individual requirements that are met using probabilistic mechanisms.

The Common Criteria, version 2, requires that the Security Target make an SOF claim of Basic, Medium, or High. No definition of these categories is given. The evaluation of this claim is based on the following definitions.

- SOF-basic: the probabilistic mechanisms have no more strength than the strength of a user-selectable (that is, non-random) numeric code or password that is less than seven characters in length. For example, a typical Automatic Teller Machine Personal Identification Number of four digits has a one-time probability of being guessed once in ten thousand attempts. Several passwords from a small table (20 to 30 entries) of passwords of at most six user-selectable characters can usually be guessed in less than a million attempts. (The estimation assumes three to four bits of randomness per character, which is consistent with dictionary attacks on passwords generally being successful with a 500,000 word dictionary.) Thus, if a probabilistic mechanism can be defeated with a probability greater than one in a million, it can only meet SOF-basic.

- SOF-medium: the probabilistic mechanisms have a strength that can be defeated with a probability significantly less than one in a million. The requirement, "significantly less" should be met by a two orders of magnitude difference, for example, one in 100 million.
- SOF-high: no definition is given, because the Security Target does not claim this.

These definitions deliberately leave a gray area, where the difference between SOF-basic and SOF-medium is debatable, but also are intended to provide the basis for stating that certain cases are clearly distinguishable.

The product clearly meets the claim of SOF-medium. Authentication, encryption, and initial integrity checking are provided by Fortezza Card services. No analysis of the strength of these services should be required as the U.S. Government promotes their use for military secret and below communications. However, the use of 80-bit keys for encryption and 160-bit hashes for integrity and authentication indicate the probability of defeating these mechanisms as many orders of magnitude less than one in a billion. (It may be as low as one in 2^{80} or approximately one in 1,000,000,000,000,000,000,000,000,000.)

The only other probabilistic mechanism is the use of a 32-bit checksum for providing integrity on user data after two Dragonfly Guards establish a secure channel using Fortezza services. The checksum is very similar to the Internet checksum, used for automatic integrity checking of all IP datagrams. The most significant difference is the Dragonfly checksum is 32 bits long instead of 16.

The checksum is calculated over the user data of the IP datagram, appended to the end of the data, and then the original data plus the checksum is encrypted using the Fortezza service in Cipher Block Chaining (CBC) mode. When the datagram is received, it is decrypted, the last 32 bits are stripped off, and the checksum is calculated to verify integrity. If the result is not the same as the stripped-off quantity, then an error message is sent that modification has been detected.

This is done on a per datagram basis. Unless an attacker can successfully decrypt, that is, has discovered the Fortezza key, the result of modifications to a captured datagram cannot be predicted. The most an attacker can accomplish is to introduce random, undetected changes into the user data. The probability of these changes being undetected is one in more than four billion or 2^{32}.

Thus, all probabilistic mechanisms for providing security services required by the Security Target meet the claim of SOF-medium.

9.3 TOE Security Assurance Requirements

Class Configuration management		
1	ACM_CAP.2	Configuration items

Yes	ACM_CAP.2.1D	The developer shall provide a reference for the TOE. *The vendor has stated that release 3.0 (as shown by output to the audit catcher on Guard startup) is the TOE. We have looked at tests for three builds, including the final build for the evaluation, 980908.1509*
Yes	ACM_CAP.2.2D	The developer shall use a CM system. *Confirmed by reference to DF_CM.*
Yes	ACM_CAP.2.3D	The developer shall provide CM documentation. *The CM documentation is DF_CM, with the configuration list being in DF_GUM*
Yes	ACM_CAP.2.1C	The reference for the TOE shall be unique to each version of the TOE. *The Guard unit is physically labeled with the model number. Each build within a release has a separate build number, which is printed out with the release number on Guard startup. Both claims were confirmed during testing.*
Yes	ACM_CAP.2.2C	The TOE shall be labelled with its reference. *The Guard unit is physically labeled with the model number. The reference (release + build) is printed out on Guard startup. The software build is also physically present on the Ignition card.*
Yes	ACM_CAP.2.3C	The CM documentation shall include a configuration list. *The configuration list is given in section 2, page 4, of the DF_GUM. It describes what is delivered to the installer as the Dragonfly Guard.*
Yes	ACM_CAP.2.4C	The configuration list shall describe the configuration items that comprise the TOE. *The list in section 2 of the DF_GUM describes the items in enough detail to identify each.*
Yes	ACM_CAP.2.5C	The CM documentation shall describe the method used to uniquely identify the configuration items. *DF_CM does this for hardware items through model numbers or specifications and for software items through module names and creation dates.*
Yes	ACM_CAP.2.6C	The CM system shall uniquely identify all configuration items. *DF_CM does this for hardware items through model numbers or specifications and for software items through module names and creation dates.*
Yes	ACM_CAP.2.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
Class Delivery and operation		
2	ADO_DEL.1	Delivery procedures
Yes	ADO_DEL.1.1D	The developer shall document procedures for delivery of the TOE or parts of it to the user. *Done in section 2 of the DF_GUM.*
Yes	ADO_DEL.1.2D	The developer shall use the delivery procedures. *Confirmed by delivery process used for delivery of six Guards to the evaluation site.*

Yes	ADO_DEL.1.1C	The delivery documentation shall describe all procedures necessary to maintain security when distributing versions of the TOE to a user's site. *The Guard is a self-contained unit. The installation procedures give a list of required equipment and accessories. The integrity of the software is tested during the initialization process and results are shown through the LEDs (section 4.1.3 and confirmed during testing) and output to the audit catcher (confirmed by testing.)*
Yes	ADO_DEL.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
3	ADO_IGS.1	Installation, generation, and start-up procedures
Yes	ADO_IGS.1.1D	The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE. *Found in section 2 of DF_GUM and on the Installation Cards.*
Yes	ADO_IGS.1.1C	The documentation shall describe the steps necessary for secure installation, generation, and start-up of the TOE. *The information provided in section 2 of the DF_GUM and on the Installation Cards does this, as was confirmed by the evaluators during test configuration setup and vendor test reruns.*
Yes	ADO_IGS.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
Yes	ADO_IGS.1.2E	The evaluator shall determine that the installation, generation, and start-up procedures result in a secure configuration. *The evaluators used the installation and start-up procedures for the test setup in order to confirm this. No generation of software is allowed, although configuration out of some software modules was done and confirmed through the configuration output at Guard startup time.*
Class Development		
4	ADV_FSP.1	Informal functional specification
Yes	ADV_FSP.1.1D	The developer shall provide a functional specification. *Done in DF_IFS. The DF_CD gives a correspondence between the DF_IFS, the DF_HLD, and test procedures.*
Yes	ADV_FSP.1.1C	The functional specification shall describe the TSF and its external interfaces using an informal style. *The DF_IFS and the DF_CD are written in English. Section 1 of the DF_IFS describes security functions, while section 2 describes external interfaces.*

Yes	ADV_FSP.1.2C	The functional specification shall be internally consistent. *Confirmed during test coverage analysis using the DF_CD.*
Yes	ADV_FSP.1.3C	The functional specification shall describe the purpose and method of use of all external TSF interfaces, providing details of effects, exceptions and error messages, as appropriate. *Section 2 of DF_IFS provides purpose and method of use for all external TSF interfaces. Exceptions and error messages are listed in DF_GUM in section 4.1.3 (LED codes for initialization and operation), section II 2.4 (audit codes), and section 2.2 (status line codes for check-ins).*
Yes	ADV_FSP.1.4C	The functional specification shall completely represent the TSF. *Confirmed by comparing the DF_IFS contents to results shown in Table 9-1.*
Yes	ADV_FSP.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
Yes	ADV_FSP.1.2E	The evaluator shall determine that the functional specification is an accurate and complete instantiation of the TOE security functional requirements. *Done by comparing the TOE security functional requirements as listed in Table 9-1 above to the DF_IFS contents. Also done by the vendor and checked by the evaluators in DF_CD.*
5	ADV_HLD.1	Descriptive high-level design
Yes	ADV_HLD.1.1D	The developer shall provide the high-level design of the TSF. *Done in DF_HLD.*
Yes	ADV_HLD.1.1C	The presentation of the high-level design shall be informal. *The DF_HLD is written in English.*
Yes	ADV_HLD.1.2C	The high-level design shall be internally consistent. *Confirmed during the analysis for Table 9-1 above.*
Yes	ADV_HLD.1.3C	The high-level design shall describe the structure of the TSF in terms of subsystems *Section 1 of the DF_HLD describes hardware subsystems, and section 2 of the DF_HLD describes software subsystems.*
Yes	ADV_HLD.1.4C	The high-level design shall describe the security functionality provided by each subsystem of the TSF. *Shown by the vendor in DF_CD and confirmed by the evaluators during preparation of Table 9-1 above.*

Yes	ADV_HLD.1.5C	The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software. *All hardware, software, and firmware as identified in the configuration list are described in the DF_HDL. The Guard depends on the Fortezza Card for supporting protection mechanisms, including encryption services, configuration information, and user identification and authentication.*
Yes	ADV_HLD.1.6C	The high-level design shall identify all interfaces to the subsystems of the TSF. *Section 2 of the DF_HLD does this for hardware interfaces such as the PC-104 bus, and section 3 does this for software interfaces.*
Yes	ADV_HLD.1.7C	The high-level design shall identify which of the interfaces to the subsystems of the TSF are externally visible. *Section 2 of the DF_HLD describes the externally visible hardware interfaces, which are further described in DF_IFS. There are no interactive user interfaces, but LED codes, audit event codes, and configuration information written to the audit catcher at Guard startup are externally visible. Section 3 of the DF_HLD describes Guard initialization and operation including the production of this externally visible information.*
Yes	ADV_HLD.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
Yes	ADV_HLD.1.2E	The evaluator shall determine that the high-level design is an accurate and complete instantiation of the TOE security functional requirements. *Done during the analysis supporting Table 9-1 and other analysis in preparation of this FER.*
6	ADV_RCR.1	Informal correspondence demonstration
Yes	ADV_RCR.1.1D	The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided. *The DF_CD provides an analysis of correspondence among the DF_IFS, the DF_HLD, and the DF_TPROC.*
Yes	ADV_RCR.1.1C	For each adjacent pair of provided TSF representations, the analysis shall demonstrate that all relevant security functionality of the more abstract TSF representation is correctly and completely refined in the less abstract TSF representation. *Confirmed by the analysis to support Table 9-1 above and also section 7 of this FER.*
Yes	ADV_RCR.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
Class Guidance Documents		
7	AGD_ADM.1	Administrator guidance

Yes	AGD_ADM.1.1D	The developer shall provide administrator guidance addressed to system administrative personnel. *The installer of the Guard is the administrator, and guidance addressed to the installer is found in section 2 of DF_GUM.*
Yes	AGD_ADM.1.1C	The administrator guidance shall describe the administrative functions and interfaces available to the administrator of the TOE. *Administrative functions for the Guard center around physical protection, correct installation, and audit review. These topics are covered in section 2 and section II of the DF_GUM. No administrative or other interactive interfaces are available to the Guard software.*
Yes	AGD_ADM.1.2C	The administrator guidance shall describe how to administer the TOE in a secure manner. *Administrative functions for the Guard center around physical protection, correct installation, and audit review. These topics are covered in section 2 and section II of the DF_GUM.*
Yes	AGD_ADM.1.3C	The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment. *Administrative functions for the Guard center around physical protection, correct installation, and audit review. These topics are covered in section 2 and section II of the DF_GUM. No software functions or privileges are configurable in the TOE.*
Yes	AGD_ADM.1.4C	The administrator guidance shall describe all assumptions regarding user behaviour that are relevant to secure operation of the TOE. *No software functions or privileges are configurable in the TOE. Assumptions regarding physical protection and correct installation are described in sections 2 and 4 of the DF_GUM.*
Yes	AGD_ADM.1.5C	The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate. *No software functions or privileges are configurable in the TOE. Section 2 of the DF_GUM contains a warning about incorrect connection of the Guard to networks, and section 4 contains troubleshooting information.*

Yes	AGD_ADM.1.6C	The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF. *No software functions or privileges are configurable in the TOE. Section II of the DF_GUM describes use of the audit catcher.*
Yes	AGD_ADM.1.7C	The administrator guidance shall be consistent with all other documentation supplied for evaluation. *Confirmed by the evaluators during test configuration setup and rerun of vendor tests.*
Yes	AGD_ADM.1.8C	The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator. *Done in section I of the DF_GUM, consisting of section 1, introduction; section 2, installation; section 3, operation; and section 4, trouble shooting.*
Yes	AGD_ADM.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
8	AGD_USR.1	User guidance
Yes	AGD_USR.1.1D	The developer shall provide user guidance. *The "user" for this TOE is the Guard identity contained in the Fortezza Card User certificate, configuration certificate, and other certificates. No interactive interface is provided. The DF_GUM provides, in section II, information required for successful configuration of the Fortezza Card, which is done by the Administration System outside of the evaluated configuration.*
Yes	AGD_USR.1.1C	The user guidance shall describe the functions and interfaces available to the non-administrative users of the TOE. *As above, there are no non-administrative human users of the Guard.*
Yes	AGD_USR.1.2C	The user guidance shall describe the use of user-accessible security functions provided by the TOE. *There are no human-user-accessible security functions provided by the TOE.*

Yes	AGD_USR.1.3C	The user guidance shall contain warnings about user-accessible functions and privileges that should be controlled in a secure processing environment. *There are no human-user-accessible security functions provided by the TOE.*
Yes	AGD_USR.1.4C	The user guidance shall clearly present all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment. *There are no human-user-accessible security functions provided by the TOE. Administrative responsibilities such as physical protection and audit interpretation have been described above.*
Yes	AGD_USR.1.5C	The user guidance shall be consistent with all other documentation supplied for evaluation. *Under the constraints described immediately above, confirmed by the evaluators during test configuration setup and rerun of vendor tests.*
Yes	AGD_USR.1.6C	The user guidance shall describe all security requirements for the IT environment that are relevant to the user. *There are no human-user-accessible security functions provided by the TOE. The security requirements for the IT environment relevant to the administrative user are described in DF_GUM, sections 2 and 4.*
Yes	AGD_USR.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
Class Tests		
9	ATE_COV.1	Evidence of coverage
Yes	ATE_COV.1.1D	The developer shall provide evidence of the test coverage. *Done in DF_CD.*
Yes	ATE_COV.1.1C	The evidence of the test coverage shall show the correspondence between the tests identified in the test documentation and the TSF as described in the functional specification. *The DF_CD provides this. The correspondence was analyzed by the evaluators and tests were added when coverage was not complete.*
Yes	ATE_COV.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
10	ATE_FUN.1	Functional testing

Yes	ATE_FUN.1.1D	The developer shall test the TSF and document the results. *Done and results provided to the evaluators as documented below.*
Yes	ATE_FUN.1.2D	The developer shall provide test documentation. *Done in DF_TPROC.*
Yes	ATE_FUN.1.1C	The test documentation shall consist of test plans, test procedure descriptions, expected test results and actual test results. *The first three are provided in DF_TPROC. The actual test results are described in 1.5C below.*
Yes	ATE_FUN.1.2C	The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed. *Test plans were added to the final version of DF_TPROC to accomplish this.*
Yes	ATE_FUN.1.3C	The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests. *The test procedures documented in DF_TPROC that apply to the evaluated configuration are documented in section 7 of this FER.*
Yes	ATE_FUN.1.4C	The expected test results shall show the anticipated outputs from a successful execution of the tests. *Each test procedure includes expected test results for each step.*
Yes	ATE_FUN.1.5C	The test results from the developer execution of the tests shall demonstrate that each tested security function behaved as specified. *The actual test results comprise an annotated copy of the test table, with handwritten notes as to outcome and any unexpected behavior; files from the audit catcher capturing Guard initialization and all audit events for the test for each Guard involved; and NetXray snoop output when it was part of the test. The complete set of actual test results examined by the evaluators was against release 2.4, build 980825.0035, and release 3.0, build 980908.1509. The evaluators confirmed that the expected results were obtained.*
Yes	ATE_FUN.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
11	ATE_IND.2	Independent testing – sample
Yes	ATE_IND.2.1D	The developer shall provide the TOE for testing. *Done. See section 7 of this FER for description of evaluator's test configuration.*
Yes	ATE_IND.2.1C	The TOE shall be suitable for testing. *Confirmed by actual test results.*

Yes	ATE_IND.2.2C	The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF. *The developer provided six Guards and accessories in order to reproduce the vendor's test configuration as closely as possible. The vendor also provided extensive developer support during evaluator testing.*
Yes	ATE_IND.2.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
Yes	ATE_IND.2.2E	The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified. *Done for the three builds received including final, as documented in section 7 above.*
Yes	ATE_IND.2.3E	The evaluator shall execute a sample of tests in the test documentation to verify the developer test results. *Done for three builds received including final, as documented in section 7 above.*
Class Vulnerability Assessment		
12	AVA_SOF.1	Strength of TOE security function analysis
Yes	AVA_SOF.1.1D	The developer shall perform a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim. *The ST makes a strength of function claim of medium but does not specify numerical values. The TOE depends on the Fortezza Card for cryptographic services. It is assumed that the Fortezza Card meets a strength of function requirement of medium for the cryptographic services that it provides. The TOE also depends on the 32-bit checksum as documented and analyzed in 98-019A. See section 9.2.*
Yes	AVA_SOF.1.1C	For each mechanism with a strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the minimum strength level defined in the PP/ST. *As above.*
Yes	AVA_SOF.1.2C	For each mechanism with a specific strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the specific strength of function metric defined in the PP/ST. *As above.*
Yes	AVA_SOF.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*

Yes	AVA_SOF.1.2E	The evaluator shall confirm that the strength claims are correct. *As above.*
13	AVA_VLA.1	Developer vulnerability analysis
Yes	AVA_VLA.1.1D	The developer shall perform and document an analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP. *Documented in DF_VA and 98-020.*
Yes	AVA_VLA.1.2D	The developer shall document the disposition of obvious vulnerabilities. *Section 2 of the DF_VA discusses the basis of trust for the TOE. Section 4 of the DF_VA discusses threats and counters to threats. Section 5 discusses remaining vulnerabilities. 98-020 addresses write-down vulnerabilities.*
Yes	AVA_VLA.1.1C	The documentation shall show, for all identified vulnerabilities, that the vulnerability cannot be exploited in the intended environment for the TOE. *Section 5 of the DF_VA discusses conditions under which the identified vulnerabilities are countered. See also sections 4.1 and 4.2 of this FER.*
Yes	AVA_VLA.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *As above.*
Yes	AVA_VLA.1.2E	The evaluator shall conduct penetration testing, building on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed. *Done. See description of independent testing in section 7 of this FER.*

Table 9-1 EAL2 Assurance Components

10 Evaluator Comments/Recommendations

None.

11 Annexes

Annex A: Dragonfly Guard Administration User Manual

The Dragonfly Guard Administration User Manual (DF_AUM), Version 2.02, dated June 1998, provides guidance on using the Dragonfly Administration System. It begins by describing the Dragonfly Communication Suite (including the Dragonfly Guards and the Dragonfly Administration System), and then describes in detail the installation and use of the Dragonfly Administration System, which is also called the Local Authority. The Dragonfly Administration System is not part of the evaluated configuration.[5]

The administrative functions and interfaces available to the administrator include management of Deployments consisting of Domains, Guards, and Hosts. For each element of the Deployment, the security parameters under control of the administrator include configuration settings as local privilege vectors, proxy RARP enabled, Audit Catcher Configured, editing of audit masks, etc.

These configuration settings are used when Fortezza cards are written using the Administration System. The administrator uses the configuration settings written into the Fortezza card to administer each Dragonfly Guard unit in a secure manner. The security impact of each configuration setting is explained and warnings about functions and privileges that should be controlled are included. This information supports the secure usage assumption found in the DF_ST applying to administrators:

A.ADMIN	The Local Authority is trusted to correctly configure User Fortezza Cards.

Section II of the DF_AUM is identical to Section II of DF_GUM, and has been discussed in section 6 above.

By combining the information in both sections, the DF_AUM describes all assumptions regarding user behavior that are relevant to secure operation of the Dragonfly Guard unit, and describes all security-relevant events relative to administrative functions. The information in both sections of the DF_AUM has been found to be consistent with the information in other Dragonfly documents furnished to the evaluators

12 Security Target

See attached document.

13 Glossary

ARP	Address Resolution Protocol
CBC	Cipher-Block Chaining
CC	Common Criteria for IT Security Evaluation
CM	Configuration Management
CPU	Central Processing Unit
CRL	Certificate Revocation List
DAC	Discretionary Access Control
DNS	Domain Name System
DSA	Digital Signature Algorithm
EAL	Evaluation Assurance Level
FTP	File Transfer Protocol
ICMP	Internet Control Message Protocol
ID	Identification
INE	In-line Encryption
IP	Internet Protocol
IT	Information Technology
IWG	Internet Gateway
KEA	Key Exchange Algorithm
LAN	Local Area Network
MAC	Mandatory Access Control
MLS	Multi-Level Secure
NSA	National Security Agency
PC	Personal Computer
PCMCIA	Personal Computer Memory Card International Association
PIN	Personal Identification Number
PP	Protection Profile
PUD	Protected User Datagram
RARP	Reverse Address Resolution Protocol
SBU	Sensitive But Unclassified
SF	Security Function
SFP	Security Function Policy
SHA	Secure Hash Algorithm
SMTP	Simple Mail Transfer Protocol
ST	Security Target
TCP	Transport Control Protocol
TNS	Tactical Name Server
TOE	Target of Evaluation
TPN	Tactical Packet Network
TSC	TSF Scope of Control
TSF	TOE Security Functions
TSP	TOE Security Policy

14 Bibliography

14.1 Dragonfly Documents

DF_AUM	ITT Industries, *The Dragonfly Administration User Manual*, Version 2.02, June 1998.
DF_CD	ITT Industries, *Dragonfly Guard Informal Correspondence Demonstration*, Version 1.4, 29 September 1998.
DF_CM	ITT Industries, *Guard TOE Configuration Management*, Reference 98-016f, 22 October 1998.
DF_GUM	ITT Industries, *The Dragonfly Guard User Manual*, Version 2.02, August 1998.
DF_HLD	ITT Industries, *Dragonfly Descriptive High Level Design Document*, Version 1.4, 15 October 1998.
DF_IFS	ITT Industries, *Dragonfly Guard Informal Functional Specification*, Version 1.2, 22 October 1998.
DF_ST	ITT Industries, *Dragonfly Guard Security Target*, Version 1.8, 29 October 1998.
DF_TPROC	ITT Industries, *Dragonfly Test Plan/Procedures*, Version 2.5, 4 September 1998.
DF_VA	ITT Industries, *Vulnerability Analysis of the Dragonfly Guard*, Version 2.4, 22 July 1998.
98-018	ITT Industries, Dragonfly Guard Configuration Files, 25 August 1998
98-019A	ITT Industries, Dragonfly 32-bit Checksums, 3 September 19998
98-020	ITT Industries, Dragonfly Anticipated Messages, 2 September 1998

14.2 Government Documents

CCITSE	*ISO 15408, Common Criteria for Information Technology Security Evaluation*, CCIB-98-026, Version 2.0, May 1998.
ST_Guide	Donaldson, Murray G., *Guide for the Production of PPs and STs*, Version 0.6, 8 July 1998, ISO/IEC JTC 1/SC 27/WG 3 N452.
Fortezza	National Security Agency, Workstation Security Products, *Fortezza Application Implementors Guide*, Revision 1.52, 5 March 1996.