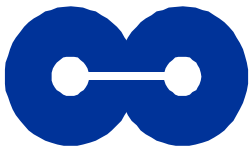
FOR PUBLIC RELEASE

DRAFT

ITT INDUSTRIES
DRAGONFLY COMPANION
FINAL EVALUATION REPORT
VERSION 2.0

Savith Kandala
Santosh Chokhani
Edward Browdy
Michael Boberski
Kristina Rogers

October 29, 1999



CYGNACOM SOLUTIONS

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

TABLE OF CONTENTS

TABLE OF TABLES AND FIGURES

1 Executive Summary

The ITT Dragonfly Companion is a network security software product that uses National Security Agency (NSA) Fortezza Cards to provide multi-level secure (MLS) services to Internet Protocol (IP) networks. Dragonfly Companions allow users to send classified or commercial proprietary information over any IP based network without worrying about that information being available to anyone other than the intended recipient. Dragonfly Companions operate on standard IP datagrams. Dragonfly Companion software is installed on a host Personal Computer (PC) that has Windows 95 as its operating system. The host PC must have a PCMCIA slot and a PCMCIA card reader. Dragonfly Companions require a PCMCIA User Fortezza Card to operate. The User Fortezza Card contains the configuration information to use the Companion. The User Fortezza Card contains nine 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. Three are used to sign and verify other certificates: the local authority, the root, and the root authority certificates. The Dragonfly Companion uses the User Fortezza Card for hashing, digital signatures, key generation, and encryption. The Companion Softkey Certificate is signed at the factory with the software authority in order to prevent pirating Companions, but it is not security relevant.

A Dragonfly Companion separates two Dragonfly Domains. In general, a Dragonfly Domain is a set of computers that are networked together without any intervening Dragonfly Units. The exception is Domain 0; this is a pseudo-domain that can be specified as the domain for the local port of more than one companion (e.g., Companion PC hosts). Computers in the same domain may be PCs, Workstations, or Servers that are all at the same security level.

The Dragonfly Administration System is used to define Dragonfly Domains and their properties. Initially, there is one Dragonfly Domain. The first Dragonfly Companion (or Guard) defined creates two domains: the Local Domain and the Remote Domain. For the Companion, the PC it protects is the local Domain, and its one ethernet interface is connected to the remote Domain. A security level is set for each Domain and these security levels may be different.

The Dragonfly Administration System is used to set the security and network configuration information. It is used to write the information onto the Companion User Fortezza Card. The Administration System requires a Local Authority Fortezza Card to create valid Companion User Fortezza Cards. The Local Authority Card is provided by ITT. The Administration System uses a graphical display and wizards to assist in the organization of a Dragonfly Deployment, a set of Dragonfly Domains. The Dragonfly Companion depends upon the Dragonfly Administration System to correctly configure its User Fortezza Card. The configuration can be verified anytime by the user using the Companion User Interface and by the Local Authority on the Administration System. The Dragonfly Administration System is outside the TOE and is considered part of the environment for the Dragonfly Companion.

Dragonfly Guards and Dragonfly Companions are collectively referred to as Dragonfly Units, but the Dragonfly Guard and Dragonfly Companion are not the same. The main differences between the Companion and Guard are as follows:

- The primary objective of the Companion is to protect the single host that it resides on from the unsecured network; the Guard is intended to protect networks of multiple hosts.
- The human user of the Companion is trusted, whereas there is no human user of the Guard.
- The local authority can configure the Companion User Fortezza Card with one or more of the options: Allow Pass Through, Firewall Mode, and Allow User to Change Default Mode. The options configured on the User Fortezza Card determine which of the following modes the user is allowed to select: Block all, Pass All, Intermediate Protection, or Firewall Protection. Table 4-2 gives a detailed description on how the configuration options and the modes of operation are related.
- On the Companion, a user must successfully login to a Fortezza Card using the correct PIN in order to use Fortezza services. The Guard's software automatically logs in to the Fortezza card.
- The Companion does not generate proxy Address Resolution Protocol requests and responses (ARPs), whereas the Guard does.
- The Companion cannot be an Audit catcher, whereas the Guard can. The Audit Catcher collects the audit data and sends updated Certificate Revocation Lists (CRLs), Audit Masks, and Routing Certificates to the Dragonfly Units for which it is serving as an Audit Catcher.

The ITT Dragonfly Companion provides the following security services: source authentication, mandatory access control, discretionary access control, confidentiality, integrity, and audit. Dragonfly Companion establishes Associations with other Dragonfly Units to authenticate each other, exchange security parameters, and establish a trusted session for communication. Dragonfly Companions use the Fortezza card to generate and securely exchange a symmetric encryption key. Dragonfly Companions and Dragonfly Units 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 is formed, messages are encrypted with a symmetric key known only to the source and destination Dragonfly Companion or Guard. From a security policy perspective, the user on the Dragonfly Companion is the user operating the Dragonfly Companion host who has the User Fortezza Card. The Dragonfly Companion identifies and authenticates itself to other Dragonfly Units based on the identity associated with the User Certificate on their User Fortezza Card. The Dragonfly Companion maintains two roles: User and Trusted Human user. The User is represented by the User Certificate on the Companion Fortezza Card and the Trusted Human User is the one who logs on to the Companion Fortezza Card with a PIN.

The Dragonfly Companion supports Mandatory Access Control (MAC) by labeling every IP Datagram with an appropriate security level and then checking that label against the security level of the destination domain before releasing the underlying datagram in plain text form. Through the sharing of security related information via an Association, Dragonfly Companions 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. The Dragonfly Companion 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 feedback for only the subset of IP based functionality for which the Dragonfly Companion 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 Companion based upon the Write Up FTP or Simple Mail Transfer Protocol (SMTP) command. If the actual message matches the predicted message, the predicted message is released. Otherwise, no message is released and there is no feedback.

The Dragonfly Companion uses Privilege Vectors for Discretionary Access Control (DAC) between Dragonfly 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. When a new Companion user Fortezza Card is being configured at the Administrative System, the Local Authority can enter privileges (for other remote domains) for the local privilege vector. This enables the Companion to communicate with other hosts in those domains.
The local authority can configure a Companion so that its local side does not represent a unique domain. In such a case, the Companion is said to be a member of the pseudo domain. The pseudo domain is also known as Domain 0, because it is represented by bit 0, the first bit, in the privilege vector.

There are three ways in which the local authority can enable communications between a Companion in the pseudo domain and a host in a real domain. They are as follows:
1. Set the privilege bit for the real domain of which the host is a member in the local privilege vector of the Companion in the pseudo domain;
2. Set all the privilege bits in the local privilege vector of the Companion in the pseudo domain; or
3. Set all privilege bits in the local privilege vector of the Dragonfly Unit protecting the real domain.

There are two main differences between the pseudo domain and real domains. First, the pseudo domain is not unique. It does not meet the definition of a domain in that there are no intervening Dragonfly Units between Companions in the pseudo domain. For this reason, companions that are members of the pseudo domain are also referred to as domainless companions.

Second, there is no interface for the local authority to set just bit 0 for the pseudo domain when programming the User Fortezza Card for the Dragonfly Unit protecting a real domain. Bit 0 is set only if all the bits are set. Therefore, if the local authority wants to control communications at a finer level of granularity, the local authority will have to reprogram the User Fortezza Cards both for the companions in the pseudo domain, if a real domain is added later. If both domain were real domains, only the User Fortezza Card for the new Dragonfly Unit protecting the new real domain would have to programmed.

The Dragonfly Companion 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 itself and other Dragonfly Units. The Companion uses the Cipher Block Chaining CBC-64 mode of operation and the Skipjack algorithm on the User Fortezza Card.

The Dragonfly Companion checks for integrity of both User Data and Dragonfly control information when messages are transmitted between itself and other Dragonfly Units. 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.

Any Dragonfly Companion can generate and send audit reports to an Audit Catcher. The Dragonfly Companion depends upon the Dragonfly Guard, which has already completed its EAL2 evaluation, to serve as its audit catcher. Audit Catchers receive audit reports from other Dragonfly Companions (and Guards) and send all messages to their serial port for printing, storage or subsequent analysis. The selection of auditable events can be controlled. The audit catcher can also send messages to a Dragonfly Companion to update the CRLs, routing certificates and audit masks.
Besides providing security services, the Dragonfly Companion allows for four modes of operation. Each mode of operation is described below and any of these modes can be chosen depending on how the User Fortezza card is configured. The local authority can configure the Companion User Fortezza Card with one or more of the options: Allow Pass Through, Firewall Mode and Allow User to Change Default. Depending on what options are configured on the User Fortezza Card, the user can choose among the following modes: Block All, Pass All, Intermediate Protection and Firewall Protection.

Block All: This mode stops the passage of all network packets to or from the Companion host system. When the Companion User Fortezza Card is removed or when a user logs out, the Companion will default to this mode, unless the default mode has been changed to Pass All in which case the Companion will default to the Pass All mode

Pass All: This mode allows free network communication with all hosts and provides no security protection. In this mode, the Companion is still running, but its security features are disabled. This mode is not permitted in the evaluated configuration.

Intermediate Protection: This mode allows for network communication at the same security level with native (i.e., non-Dragonfly) hosts in its remote domain, but uses Dragonfly encryption to communicate with other Dragonfly Units. This mode cannot be reached when the trusted human user is not logged in.

Firewall Protection: This mode allows network communication with other Dragonfly Units only, and all the TOE security functionality is enabled. This mode cannot be reached when the trusted human user is not logged in.
Dragonfly Companions do not have to be programmed with complete deployment information as they use a trusted, automatic discovery mechanism to learn the system topology. Dragonfly Companions allow use of Internet Control Message Protocol (ICMP) messages, ICMP Echo Requests (pings) and ICMP Echo Responses to find out in which Dragonfly Domain a destination host is located. The ICMP Echo Request is transmitted at the same time as an Association Request. Once the Dragonfly Domain of the host is located, the source and destination Dragonfly Companions can exchange security levels and generate a symmetric key for encryption. Neither the initiating Dragonfly Companion nor the destination Dragonfly Unit needs to know the name, address, or even the existence of the other prior to the Association setup. Once the Association is set up, both Dragonfly Units know all that they need to know.

The Dragonfly Companion provides in-line encryption (INE) functionality to tunnel data through a network at a different security level. Dragonfly Companions allow hosts at a lower security level to send communications through a network at a higher security level to another host at the same lower security level as the original host. Higher level information is not released to the lower level hosts. For example, two hosts at the SBU level could tunnel data through a Secret network. In addition, hosts at a higher security level can communicate over a network at a lower security level without releasing information from the higher security level to the lower security level. For example, two hosts at the Sensitive but Unclassified (SBU) level could tunnel data through an unprotected Unclassified network. When two or more Dragonfly Units exist along a data path, they provide confidentiality, integrity, and source authentication.

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 host PC. 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 Companion against the Security Target as authorized NSA under its Trust Technology Assessment Program. The Dragonfly Companion product satisfies the functional and assurance requirements of its Security Target and therefore should be awarded a certificate at EAL2.

2 Identification

The Target of Evaluation (TOE) consists of:

- ITT Industries Dragonfly Guard Model G1.2 running Dragonfly software release 3.0, Build 980908.1509.
- Windows 95 Operating System, and
- ITT Dragonfly Companion, Version 3.02, Build 129.

2.1 Dragonfly Guard

The ITT Dragonfly Guard configured to serve an Audit Catcher is part of the TOE. The ITT Dragonfly Guard has completed its EAL2 evaluation. The Companion relies on the Guard configured as an audit catcher to receive its audit records. In addition, the Guard sends messages to the Companion to update its Certificate Revocation List, Routing Certificate and Audit Mask. The security functionality provided by the Guard is documented in the ITT Industries Dragonfly Guard Security Target [DF_GST] and evaluation report is provided in the ITT Industries Dragonfly Guard Final Evaluation Report [DF_GFER].

The Dragonfly Companion relies on the Guard serving as its Audit Catcher to receive updated CRLs, Audit Masks and Routing Certificates. For this reason the Dragonfly Guard was included in the TOE.

2.2 Windows 95

The Dragonfly Companion resides within the Windows 95 operating system as a Virtual Device Driver (VxD).

The Companion also relies on Windows 95 operating system to provide domain separation and non-bypassability of the TSP. For this reasons Windows 95 was included in the TOE.

2.3 Dragonfly Companion

The Dragonfly Companion Version 3.02, build 129 is installed in a personal computer with in Intel CPU running Microsoft Windows 95.

2.3.1 Hardware

The PC running Dragonfly Companion has the following hardware requirements:

- External Ethernet
- PCMCIA Reader
- FORTEZZA Card

2.3.1.1 External Ethernet

The Companion host PC must have an ethernet network interface for the Companion's Remote Port. Otherwise there is no need for the Companion.

2.3.1.2 PCMCIA Reader

The PC must have at least one available PCMCIA reader slot to allow it to access the Fortezza card. The reader is not security relevant, because it does not implement any TOE Security Functions. If the PCMCIA Reader does not function correctly then the Companion cannot form an Association with other Dragonfly Units.

2.3.1.3 Fortezza Card

The Fortezza Card provides all cryptographic services used by Dragonfly Units. The Fortezza Cards implement SKIPJACK symmetric encryption and decryption, Secure Hash, Digital Signature, Key Generation, and Key Exchange. The FORTEZZA cards also store keys and certificates.

2.3.2 Software

The Dragonfly Companion was designed to include the following major software partitions:

- Dragonfly Initialization
- PCMCIA Card Interface
- User Interface

- Network (NDIS) interface

2.3.2.1 Dragonfly Initialization

Initialization of the Dragonfly Companion is accomplished in several steps. The first step is accomplished when the software is started by Windows. At this time the Companion driver performs numerous initialization functions which include:

- Finding and binding to the Microsoft TCP/IP protocol stack (95 version)
- Allocating memory and initializing data structures.
- Initializing buffers for Configuration information and writing the Companion configuration data to these buffers from the Fortezza card. Verifying that the information is consistent with the Administration system. (Verifying the Dragonfly Configuration Certificate on the Fortezza Card.)
- Setting the Dragonfly code's internal time. (The absolute time is taken from the user Fortezza Card to set the Companion's internal time initially. The system clock of the host PC is used to increment the Companion's internal time. This is done by calculating the actual time since boot (number of ticks on the clock). At regular intervals this time is checked against the time on the fortezza card.

Note: The actual time since boot cannot be set by the user which is the number of ticks (seconds) on the host PC clock since the host PC bootup.)

- Starting the main event timer
- The Companion listens in on the outgoing traffic to determine its own IP address. Until the IP address is received, the Companion is in a hold state (IP_HOLD). This means that all IP traffic will be blocked in Intermediate or Firewall mode until the IP address is determined (although non-IP packets still go through until the IP address of the host PC is determined.

All other initialization of the software happens when a Fortezza card is inserted and when a user logs in to the Companion. All memory used by the Dragonfly Companion driver (NDISTRAP.VXD) for encrypted or for unencrypted network packet storage is allocated at start-up and is static. There is no memory allocation for network packet data done during normal operation. There are two major memory heaps,

1. The Trusted Packet Memory
2. The Trusted Carrier Memory.

2.3.2.2 PCMCIA Card Interface

The PCMCIA interface in the Companion has two major tasks:

1: When a Fortezza card is installed in the PC, the PC Card driver software in the operating system sends the Companion driver a Card Insertion event. At this time the Companion does the following:

- Maps and verifies proper mapping of the Fortezza shared memory.
- Basic initialization of the Fortezza card.
- Check the Fortezza card to see if it is a Companion card and not a Local Authority card or a Guard card. The "Softkey" Certificate is used to verify that the card is a Companion card.
- Read the time from the Fortezza Card and set the Companion's internal time.

2: When a user attempts to log in to the Companion software he or she enters their user PIN through the User Interface. The user PIN is assigned by the Local Authority while configuring the user Fortezza Card. At this time the Fortezza initialization enables the cryptographic services on the card using the supplied Fortezza User Pin. The software also verifies the operation of the card. Each of the certificates stored on the card is validated using the Secure Hash and Digital signature functions of the Fortezza Card.

The Fortezza initialization code also schedules a task to locate and check-in with an audit catcher, if an audit catcher is specified in its configuration certificate. An Audit Catcher is a Dragonfly Guard that has been designated by the Local Authority to collect and record Audit Messages from other Dragonfly units. (A Companion cannot be configured to be an audit catcher). As part of the check-in process, the Dragonfly unit discloses the revision numbers of its current Revoked Certificate List, Audit Event Mask, and Routing Certificate. (The use of these certificates is described later). If any of these certificates is older than the ones maintained by the Audit Catcher, a new version of the certificate(s) is sent back to the Dragonfly unit where it is cryptographically validated and stored in the Fortezza Card. If the unit is configured to "Require an Audit Catcher", it will not begin to process user data until the more recent versions of certificates have been received. If an Audit Catcher is not required, the unit will continue to attempt to obtain new certificates, but will process user data. Dragonfly units check-in with their audit catcher periodically at an interval determined by the Local Authority and the validity of the unit's certificates are checked each time.

2.3.2.3 User Interface

There are two main programs that comprise the user interface for the Companion:

1. A small login program is run by Windows at boot time, which allows the user to input a PIN to log in to the Fortezza Card and activate the Companion software.
2. When Windows has booted and the trusted human user has logged on to the Fortezza card, a graphical user interface program is started which allows users to interact with the Companion driver program in many ways, including the following:
   - Log in to the card & Log out. Note the login menu option is only used, if the user has previously used the log out option.
   - View a list of associations and also configuration data for the Companion
   - Choose an interface to bind to (Windows 95 version)
   - Set the Operational Mode of the Companion

2.3.2.4 Network Driver Interface Specification (NDIS) Interface

The Companion puts itself in place in a binding between the Microsoft TCP/IP stack and its network devices. All bindings of this type are broken and then the Companion connects to one of them, giving preference to an ethernet port if it exists. This binding is user-selectable, however, through the control GUI. The diagram below shows the placement of the Dragonfly Companion driver in the system.

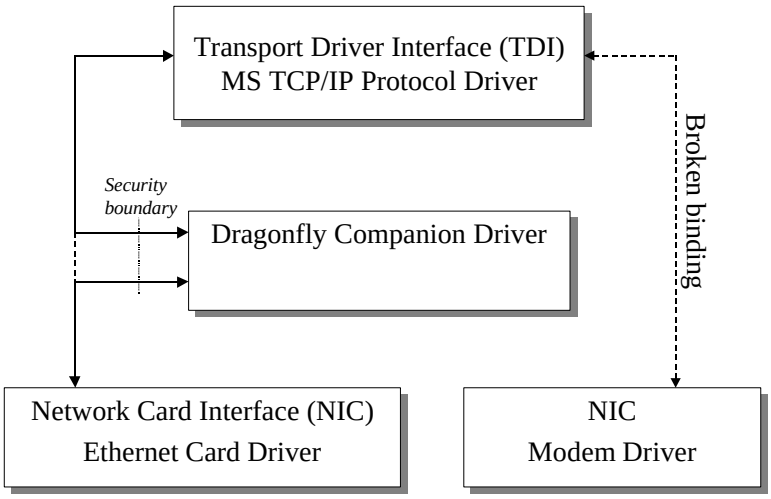


Figure 2-1 - Companion NDIS Binding in the Network

2.3.3 System Requirements

The Companion host system requirements are enumerated in the Dragonfly Companion User's Manual (DF_UM) as follows:

1. Addressing Requirements:
   The address space in the Companion host system should have sufficient space for a 16K block of memory and a 4K block of memory between memory addresses C0000 and EFFFF.

2. Operating System Requirements:
   The Companion requires a host running the Windows 95 operating system.

3. PCMCIA Card Reader:
   The Companion host system requires a PCMCIA card reader installed. Not all PCMCIA card readers are compatible with the Companion software. The user has to check with Dragonfly Technical Support regarding questions about the PCMCIA reader compatibility.

The software and hardware components of the Dragonfly Companion 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 Companion: 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.

3.1 Identification and Authentication Policy

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

A User Fortezza Card must be inserted in order for a Dragonfly Companion to start up. If the User Fortezza Card is removed, the Companion goes into either Block All or Pass All mode, depending on the configuration options. The Fortezza card contains a User Fortezza Certificate that is used to identify the User Role.

A user must successfully login to a Fortezza Card using the correct PIN in order to use Fortezza services. The user must login every time the Companion boots, when the user has been logged out or when the Fortezza Card is removed and reinserted. Dragonfly Companion User Certificates and PINS for the Companion Fortezza Card are created on an Administration System by the local authority. The local authority must enter the correct PIN for the local authority certificate in order to login to the Administration System.

Source authentication is performed when one Dragonfly Companion requests an Association with another Dragonfly unit. The source Dragonfly Companion digitally signs the Association request and the destination Dragonfly Companion verifies the digital signature.

Before an Association Request is granted, checks are made to ensure that the certificate of the requesting Dragonfly Unit has not expired and is not on the Certificate Revocation List. The Certificate Revocation List is updated via the Audit Catcher. The details are described in Section 3.4, Audit Policy.

3.2 Mandatory Access Control Policy

Dragonfly units prevent User Data from passing between one Dragonfly Domain and another unless the destination domain's security level dominates that of the originating Domain (i.e. Mandatory Access Control, MAC). Dragonfly accomplishes this by labeling all packets it protects with a Security Level of Unclassified, Sensitive but Unclassified, Confidential, Secret or Top Secret. The Security label is within the "signed data envelope" The Packet Security Level is set equal to the security level of the Dragonfly Unit's port on which the packet was received. Once a 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. The MAC rules are then

applied based on the comparison. Even if the MAC rules are satisfied, some packets may not be released based on Discretionary Access Control.

3.2.1 Mandatory Access Control Policy Statement.

A Dragonfly Companion will not release IP Datagrams containing User Data from a domain at a higher security level to a domain at a lower security level. Dragonfly Companions implement the following security levels:

Unclassified,
Sensitive but Unclassified,
Confidential,
Secret,
Top Secret.

The level dominance relationship between them is as follows: Top Secret strictly dominates Secret. Secret strictly dominates Confidential. Confidential strictly dominates Sensitive but Unclassified. Sensitive but Unclassified strictly dominates Unclassified.

3.2.2 Write Equal

The MAC policy imposes no restrictions on the flow of IP datagrams between Dragonfly Domains at the same level.

3.2.3 FTP Datagrams Supported for Write Up

The following File Transfer Protocol (FTP) commands are allowed, if write-ups are enabled for the Companion:

ABOR, APPE, MODE, NOOP, PASS, PORT, PWD, SRTU, STOR, STOU, TYPE, USER

3.2.4 SMTP Datagrams Blocked for Write Up

The following Simple Mail Transfer Protocol (SMTP) commands are always blocked for Write Up, even if Write Ups are enabled for the Companion:

EXPN, HELP, LIST, RETR, STAT, STT, TOP, TURN
Other SMTP datagrams are allowed if writeups are enabled.

3.2.5 Allowed Information Flows

The Dragonfly Companion can be configured to allow the following control information to be released from a higher security level Dragonfly Domain to a lower security level Dragonfly Domain (by the Local Authority with the option allow write-ups in the Dragonfly Administration System. Section 5.3.2.4 describes this in more detail):

a) ICMP responses

b) UDP and TCP Name Server responses with a single answer, and

c) Anticipated FTP or SMTP messages as described below.

No other IP datagrams are allowed to flow from a higher level Dragonfly Domain to a lower level Dragonfly Domain.

Note: Address Resolution Protocol (ARP) and Reverse Address Resolution Protocol (RARP) requests and responses pass through the Companion without being processed by it.

3.2.6 FTP and SMTP Anticipated Responses

In order for the FTP and SMTP protocols to work, it is necessary for responses to the allowed write up messages to be returned to the originating host. The Dragonfly Companion has implemented a patented write up mechanism of anticipated responses to control the information that can flow from higher level Dragonfly Domains to lower level Dragonfly Domains as responses.

When a Dragonfly Companion releases a message for write up, it creates the anticipated response at the security level of the originating host. When the Dragonfly Companion receives an actual response from the write up message, it compares it to the anticipated response. If the actual response matches the anticipated response, the anticipated response is released to the originating host. If the actual response and anticipated response do not match, nothing is released to the originating host and an audit event message may be generated. The Companion's audit events can be individually enabled or disabled via the "Audit Mask Certificate", which is set by the Local Authority. If an auditable event, which the Audit Mask has enabled, occurs, an Audit message is generated and sent to the audit catcher.

In some cases, it is necessary to copy some fields of control information (such as number of bytes received) from the actual response to the anticipated response. These copied fields allow information to flow from the higher level Dragonfly Domain to the lower level Dragonfly Domain. The fields and number of bytes that are copied from the actual response to the anticipated response are described in the document 99-004 ITT Industries, Dragonfly Anticipated Messages, 22 March 1999.

3.2.7 Name Server Requests and Responses

The Dragonfly Companion releases Name Server requests and responses without performing a mandatory access control (MAC) check. Name Server Requests are allowed from low host to high servers and Responses from high servers to low hosts only if "Write Ups" are enabled; otherwise, they are blocked. Name Server Requests from high hosts to low servers are always blocked. (This would be audited as an attempted "Write Down").

3.2.8 ICMP Requests and Responses

Dragonfly Companions allow the following ICMP requests for writeup:

ICMP Echo Request, and

ICMP Time Stamp Request.

Dragonfly Companions allow the following anticipated ICMP responses for writedown:

ICMP Echo Response,

ICMP Time Stamp Response,
ICMP Unreachable Destination,
ICMP Source Quench, and
ICMP Time Exceeded.

3.3 Discretionary Access Control Policy

In addition to providing Mandatory Assess Control based on Security levels, Dragonfly Units prevent User Data from passing between one Dragonfly Domain and another unless at least one of the Domains is explicitly given the privilege to communicate with the other (i.e. Discretionary Access Control). The User Fortezza Certificate includes information about the privileges of each domain attached to the devices' Local and Remote ports in the form of Privilege Vectors. A Port's Privilege Vector is a list of the Dragonfly Domains with which a host attached to the port can form an association. When an Association is being established, the Dragonfly software checks to see if the Dragonfly Domain that includes the destination host is contained in the Privilege Vector of the requesting Dragonfly Unit. If it is, the Association is allowed. If not, a second check is done to see if the Dragonfly Domain that includes the requesting host is contained in the Privilege Vector of the destination Dragonfly Unit. Again, if it is, the Association is allowed. Otherwise, an "Association Denied" message is sent to the originating Dragonfly Unit.

3.3.1 Privilege Vectors

Dragonfly Companions enforce DAC between the source Dragonfly domain and the destination Dragonfly domain using privilege vectors. Each domain has a privilege vector associated with it. Other Dragonfly Domains are represented by bits in the privilege vector. If either the destination domain bit is set in the source domain's privilege vector, or the source domain bit is set in the destination domain's privilege vector, an Association may be formed between hosts in the source domain and the destination domain. DAC checks are performed at the time of Association. DAC checks provide the ability to control the release of IP datagrams between Dragonfly Domains at the same security level. For the Companion, the local authority can set the local privilege vector bits for remote Domains.

3.3.2 Shared Domain

The local authority can configure a Companion so that its local side does not represent a unique domain. In such a case, the Companion is said to be a member of the pseudo domain. The pseudo domain is also known as Domain 0, because it is represented by bit 0, the first bit, in the privilege vector.
There are three ways in which the local authority can enable communications between a Companion in the pseudo domain and a host in a real domain. They are as follows:

1. Set the privilege bit for the real domain of which the host is a member in the local privilege vector of the Companion in the pseudo domain;
2. Set all the privilege bits in the local privilege vector of the Companion in the pseudo domain; or

3. Set all privilege bits in the local privilege vector of the Dragonfly Unit protecting the real domain.

There is no interface for the local authority to set just bit 0 for the pseudo domain when programming the User Fortezza Card for the Dragonfly Unit protecting a real domain. Bit 0 is set only if all the bits are set. Therefore, if the local authority wants to control communications at a finer level of granularity, the local authority will have to reprogram the User Fortezza Cards both for the companions in the pseudo domain, if a real domain is added later.

3.3.3 Block All Mode

If a Dragonfly Companion is in the Block All mode, no IP datagrams are released.

3.3.4 Pass All Mode

If a Dragonfly Companion is in the Pass All mode, the Discretionary and Mandatory access control functions of the Companion are disabled and all IP datagrams are released. (Note: This mode is not allowed in the evaluated configuration.)

3.3.5 Intermediate Protection Mode

If a Dragonfly Companion is in the Intermediate Protection Mode, the Companion releases all IP datagrams to non-Dragonfly protected hosts, if the MAC check passes. The Dragonfly Companion uses Dragonfly encryption to communicate with other Dragonfly Units, if the MAC and privilege vector (DAC) checks pass.

3.3.6 Firewall Protection Mode

If the Dragonfly Companion is in Firewall mode, then IP datagrams are released to other Dragonfly Units only, if the MAC and the privilege vector (DAC) checks pass.

3.3.7 No Native Associations Routing Option

“No Native Associations” option prevents Dragonfly Units from establishing Native associations. This option can be selected from the Dragonfly Administration system and the information is stored in the Routing Certificate on the Companion Fortezza Card. If the No Native Associations Routing Option (Type field in Routing Certificate) is set, the Companions behaves as if it is in Firewall Mode, even if it is in Intermediate Mode with respect to the associated IP address. This option is only applicable if the Companion is in Intermediate Mode.

3.4 Audit Policy

Each Dragonfly Companion reports events to an Audit Catcher, if the configuration certificate of the Companion identifies an audit catcher. The Configuration Certificate, generated by the Local Authority, contains a circular list of up to five Audit Catchers. The Companion tries to maintain an association with one audit catcher at all times. During Dragonfly initialization, an association request is sent to the first audit catcher in the list. If no association is obtained after the number of retries specified in the Configuration Certificate, the next audit catcher in the list is tried. This process continues until an association is established. If the Companion is configured to "Require Audit" by the Local Authority (by setting the "Requires Audit Catcher" option in the Dragonfly Administration System to "Yes"), then no user data processing is attempted until the Companion has successfully "Checked-in" with the Audit Catcher. The local authority can specify a list of Audit Catchers (up to five) when the Companion User Fortezza Card is configured on the Dragonfly Administration System.

A task in the permanent schedule table, sends periodic "Check-in" messages to the Audit catcher, and a response is expected. The time interval between "check-in" is configurable. The Local Authority can configure it by using the "Edit Advanced Settings" option in the Dragonfly Administration System. The Local Authority can specify the "Check in delay" in seconds. If a check-in (or a audit) message is sent the specified number of times without a response, the Companion automatically attempts to establish an association with the next Audit Catcher in the list and continues sending "Check-in" messages.

There are several other certificates written to the Fortezza card by the Local Authority that effect the operation of Dragonfly units. These certificates are updated as needed during check-in with the Audit Catcher.

Audit Mask Certificate:
This certificate specifies which of the possible audit events will be reported to the Audit Catcher.

Revoked Certificate List Certificate:
This certificate contains a list of the certificate ID of all Dragonfly user certificates issued by the Local Authority that are now revoked.

Routing Certificate:
When a Dragonfly Companion is deployed in environments where it needs to communicate with other Dragonfly units that are not routable (e.g., some IP addresses are hidden by Firewalls or gateways), the routing table contains information about what IP addresses to use when communicating with hosts in the hidden domains. The Routing Certificate is also used to prevent Dragonfly Units from establishing Native Associations with specific IP addresses.

As part of the check-in message, the revision number of the Companion's current Certificates is sent. When Check-in messages are received, the revision numbers of the Audit Mask, Revoked Certificate List, and Routing certificates are checked against the Audit catcher's current versions, and a reply sent indicating which if any certificates are out of date. Subsequent messages are sent that contain the actual updated certificates if necessary.

The Companion's audit events can be individually enabled or disabled via the "Audit Mask Certificate". If an auditable event, which the Audit Mask has enabled, occurs, an Audit message is generated and sent in a Protected Dragonfly Datagram (PDD) message. The message contains an ID number unique to the Companion, a time stamp, an audit event number, and descriptive text.

Only Guards, not Companions, can be configured by the Local Authority to serve as Audit Catchers. When operating in Audit Catcher mode, a Guard can perform all the normal Guard functions, but can also receive Check-in and Audit Messages from other Dragonfly Units. These messages are cryptographically authenticated like any other PDD message. A response is sent back to the reporting Companion, to indicate correct receipt of the Audit message. The Audit messages are then formatted to include a message ID number unique to the audit catcher, and a time stamp.

3.4.1 Audit Masks and Audit Mask Management

The Audit Mask is a 256 bit vector with one bit for each auditable event. If an event is to be audited, the bit is turned on in the Audit Mask.

The Local Authority can define an audit mask, which determine what events will be audited and sent to an audit catcher. The Dragonfly Administration System initially creates three audit masks defined as "standard", "audit all" and "audit none". When the Local Authority configures the User Fortezza Card for a Dragonfly Companion, it can select either Standard, Audit All, or Audit None.

For the standard audit mask the events to be audited can be selected by the local authority (by the Edit Audit Mask option on the Dragonfly Administration System). If the Dragonfly Companion is configured to use the "Standard" Audit Mask, then the events selected to be audited will be audited. The "Standard" Audit Mask can be updated during normal operations by the Audit Catcher. This means that the selection of auditable events can be changed during normal operations. It requires inserting an updated User Fortezza Card for the Audit Catcher and re-initializing the Audit Catcher. If the Dragonfly Companion is configured to use "Audit All" then all the events are audited, and if the Dragonfly Companion is configured to use "Audit None" then none of the events will be audited.

Audit masks are part of a Dragonfly Companion's initial configuration and are updated by the Audit Catcher. The Audit Mask is identified by name and version number. The Dragonfly Companion reports the identity of its current Audit Mask to the Audit Catcher in its Check-in Message. The Audit Catcher compares the reported Audit Mask with its current one. If the Dragonfly Companion has an out-of-date Audit Mask, the Audit Catcher sends the current Audit Mask back to the Dragonfly Companion.

4 Assumptions and Clarification of Scope

This section describes the assumptions made by the evaluation team about secure use of the Dragonfly Companion, and assumptions about the physical environment in which it 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.

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

	Assumption Name	**Assumption Description**
1	A.Attack_Level	Attackers are assumed to have a medium level of expertise, resources, and motivation.
2	A.Crypto_Services	Cryptographic services are provided by the User Fortezza Card.
3	A.Crypto_SOF	The cryptographic algorithms on the Fortezza card are assumed strong enough to counter at least a medium level of attack.
4	A. Local_Auth	The local authority is trusted to correctly configure User Fortezza Cards. In addition, the local authority is trusted to set the time correctly on the User Fortezza Cards
5	A.No_Lower_Level_Attack	It is assumed that Windows 95 cannot be attacked through lower level network protocols (i.e., below IP layer).
6	A.No_Other_Programs	No other programs may be installed on the host computer besides Windows 95 and the Dragonfly Companion.
7	A.No_Untrusted_Users	There are no untrusted users on the Dragonfly Companion
8	A.Only_One_IP_Port	The human user is trusted to configure Windows 95 so that there is only one network and it only accepts IP datagrams.
9	A.Physical	The Dragonfly Companion Host system is assumed to be protected from physical tampering.
10	A. User	The only user on the Dragonfly Companion is the trusted human user who has been provided with the user PIN for the User Fortezza card. The human user is assumed able to install the Dragonfly Companion in the evaluated configuration in accordance with the IGS Procedures. The human user is assumed able to insert the correct User Fortezza Card into the Dragonfly Companion, to connect its port to the network and to put the Companion in a proper mode. The human user is trusted not to bypass or tamper with the security enforcing functions of the Dragonfly Companion.

11	A.Windows_95	The Dragonfly Companion is installed on a Windows 95 operating system with the specified hardware configuration.

Table 4-1 Secure Usage Assumptions

4.2 Environmental Assumptions

This section documents the environmental assumptions made about the product during the evaluation. The environmental assumptions for the Dragonfly Companion are listed in section 5.3 of DF_ST. The main components in the IT Environment of the Companion are the FORTEZZA card and the Dragonfly Administration System.

The Dragonfly Companion relies upon the Fortezza Card to provide the following:
1. Cryptographic services including : secure hash, digital signature, key exchange algorithm, and symmetric key encryption.
2. Storage of certificates, private keys and symmetric keys.
3. Generation of the time stored in its audit records.

The Dragonfly Companion relies upon the Dragonfly Administration System to configure the system by setting its security attributes and creating the User Fortezza Card. The security attributes of a Dragonfly Companion are set by the local authority on the Dragonfly Administration and the User Certificate on the Dragonfly Companion's User Fortezza Card is signed by the local authority.

4.3 Clarification of Scope

This section describes the various configurations for the Dragonfly Companion. There are three configuration options related to mode on the User Fortezza Card that can be set by the local authority on the Dragonfly Administration System:

a) Pass Through Allowed

b) Firewall Mode, and

c) Allow User to Change Default.

The Companion has the following modes of operation Block All, Pass All, Intermediate Protection and Firewall Protection. These modes can be set by the trusted human user on the Dragonfly Companion depending on the configuration options on the user Fortezza card set by the Local Authority. Table 4-2 lists the possible configuration options set and the modes available to the trusted user.

There are three ways in which the trusted human user can log off the companion:
1) By selecting the Log Off option
2) By shutting down the Companion user interface, and
3) By removing the Fortezza card.

At this time the Companion will revert to its default state either Block All mode or Pass All mode. In the evaluated configuration, this will always be Block All mode. When the user logs off, the values for the three configuration options: Pass Through Allowed, Firewall Mode, and Allow User to Change Default are stored in the registry and used until a user logs in again. Only Firewall Mode can be changed in the evaluated configuration. Pass Through Allowed and Allow User to Change Default are always disabled in the evaluated configuration.

The first half of Table 4-2 shows the mode that the Companion can default to if the trusted human user is not logged in. The second half of the table shows the modes that the trusted human user can select when logged in.

If the trusted human user is not logged in, the only allowed mode is Block All mode, if the Pass Through Allowed option is not set on the Companion User Fortezza card as required in the evaluated configuration. The Companion allows Block All and Pass All mode, if the Pass Through Allowed option is configured on the Companion User Fortezza card. If the card is set with the Allow User to Change Default option, the user can set the default to Pass All mode using the Pass All Packets Before Login Entry. The Pass Through Allowed option is also disabled in the evaluated configuration.

If the trusted human user is logged in, the Companion can always be used in Block All mode or Firewall Protection mode. The Companion allows Pass All mode, if the Companion is configured with the Pass Through Allowed option or the Allow User to Change Default option and the user chooses to change the default mode. The Companion allows Intermediate Protection mode, only if the Firewall Mode option is not configured on the User Fortezza Card.

The security state menu of the Dragonfly Companion has four options: Block All Packets, Pass All Packets, Intermediate Protection, and Firewall Protection. The trusted human user can select one of these options when logged in, based on the rules depicted in Table 4-2.

If the Allow User to Change Default option is set to "Yes", the Pass All Packets Before Login Menu Entry is enabled, and the trusted human user is allowed to change the default mode before login to Pass All.

The following modes can be reached by the trusted human user:

Block All Mode: Block All mode is the default mode when the trusted human user is not logged in, unless the Pass Through Allowed option has been selected by the local authority which is not allowed in the evaluated configuration. Block All mode can always be selected by the trusted human user when s/he is logged in.

Pass All Mode: Pass All is the default mode when the trusted human user is not logged in, if the Pass Through Allowed option is selected. However, this is not allowed in the evaluated configuration. This option also allows the trusted human user to select Pass All mode when logged in. If the Allow User to Change Default option is selected, the trusted human user has access to the Pass All Packets Before Login Entry and can use this to set the default mode to Pass All. If this option is set, the trusted human user can also set the Companion to Pass All Mode when logged in.

If a Dragonfly Companion is in the Pass All mode, the Discretionary and Mandatory access control functions of the Companion are disabled and all IP datagrams are released. However, this is not allowed in the evaluated configuration. The Local Authority can disable a this mode for the Trusted Human User by not selecting (i.e. check) the "Allow Pass through mode" option or the "Allow user to change default " option while configuring the Companion Fortezza Card.

Intermediate Protection Mode: The Companion defaults to Intermediate Protection mode when the trusted human user is logged in, if the Firewall Mode option is not set. If the Companion has been set to some other mode, the trusted human user can set it back to Intermediate Protection Mode when logged in if the Firewall Mode option is not set. This mode cannot be reached when the trusted human user is not logged in to the Companion.

Firewall Protection Mode: The Companion defaults to Firewall Protection mode when the trusted human user is logged in, if the Firewall Mode option is set. If the Companion has been set to some other mode, the trusted human user can set it back to Firewall Protection Mode when logged in. This mode cannot be reached when the trusted human user is not logged in to the Companion.

Logged In	**Pass Through Allowed****	**Firewall Mode**	**Allow User to Change Default ****	**Allowed Modes 1=Block All 2=Pass All 3=Intermed. Protection 4=Firewall Protection**			
No***	No	No	No	**1***			
No++	No	No	Yes	**1***	2+		
No	No	Yes	No	**1***			
No++	No	Yes	Yes	**1***	2+		
No++	Yes	No	No	1	**2***		
No++	Yes	No	Yes	1	**2***		
No++	Yes	Yes	No	1	**2***		
No++	Yes	Yes	Yes	1	**2***		
Yes	No	No	No	1		**3***	4
Yes++	No	No	Yes	1		**3***	4
Yes	No	Yes	No	1			**4***
Yes++	No	Yes	Yes	1			**4***
Yes++	Yes	No	No	1	2	**3***	4
Yes++	Yes	No	Yes	1	2	**3***	4
Yes++	Yes	Yes	No	1	2		**4***
Yes++	Yes	Yes	Yes	1	2		**4***

* The **bold** values in the table are the modes that the software will go to by default without user reconfiguration.
** Option not allowed in the evaluated configuration.
*** This is the initial state of the Dragonfly Companion after installation.
+ The trusted human user can set the default mode to Pass All mode by making use of the Pass All Packets Before Login Entry if the Allow User to Change Default option is enabled. This state will not be reached in the evaluated configuration.

++ This state will not be reached in the evaluated configuration, because Pass Through Allowed and Allow User to Change Default must always be "no."

Table 4-2 Modes Allowed by Configuration Options

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.

The Dragonfly Administration System is outside the TOE. 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 available from the Companion main menu. (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 Guard is a network security device. It 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. The second card is a Fortezza Card with several digitally signed certificates containing network configuration information. The Dragonfly Guard is interoperable with the Dragonfly Companion. A Dragonfly Guard configured to serve as an audit catcher to receive the audit records. The audit catcher is also required to send updated certificate revocation lists and audit masks to the Dragonfly Companion.

4.3.1 Domainless Dragonfly Companions

The local authority can configure a Companion so that its local side does not represent a unique domain. In such a case, the Companion is said to be a member of the pseudo domain. The pseudo domain is also known as Domain 0, because it is represented by bit 0, the first bit, in the privilege vector.

There are three ways in which the local authority can enable communications between a Companion in the pseudo domain and a host in a real domain. They are as follows:

1. Set the privilege bit for the real domain of which the host is a member in the local privilege vector of the Companion in the pseudo domain;
2. Set all the privilege bits in the local privilege vector of the Companion in the pseudo domain; or
3. Set all privilege bits in the local privilege vector of the Dragonfly Unit protecting the real domain.

There are two main differences between the pseudo domain and real domains. First, the pseudo domain is not unique. It does not meet the definition of a domain in that there are no intervening Dragonfly Units between Companions in the pseudo domain. For this reason, companions that members of the pseudo domain are also referred to as domainless companions.

Second, there is no interface for the local authority to set just bit 0 for the pseudo domain when programming the User Fortezza Card for the Dragonfly Unit protecting a real domain. Bit 0 is set only if all the bits are set. Therefore, if the local authority wants to control communications at a finer level of granularity, the local authority will have to reprogram the User Fortezza Cards for the companions in the pseudo domain, if a real domain is added later. If both domain were real domains, only the User Fortezza Card for the new Dragonfly Unit protecting the new real domain would have to programmed.

4.3.2 Military Network Configuration Issues

The Dragonfly Companion 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.

The Tactical Packet Network (TPN) is an IP-based network used by the U.S. Army. The configuration options to support the TPN were not set during evaluation testing, and the TPN was not considered during the analysis for this report. Support for the TPN is not within the evaluated configuration (although support for DNS, SMTP, and FTP, which TPN depends upon, is part of the evaluated configuration). There is an option in the Dragonfly Administration System for Military Network Configuration. The option is “TPN/IGW support available on None/Remote”. None was selected during the evaluated configuration.

4.3.3 Routing Certificates

In order to create greater inter-activity with networks configured using Firewalls, the Dragonfly technology now includes the ability to create and edit routing tables. These tables are required to allow communication between two Dragonfly Domains where a Proxy Firewall resides between them. In this case, special routing is required to initiate an association from the outside of the firewall, and the routing tables will instruct the Dragonfly Unit how to conduct this special routing. This process also requires the Firewall administrator to program the Firewall to use a Designated Dragonfly Guard (DDF). This information is stored in the Routing Certificate on the Companion Fortezza Card. This certificate contains information about IP address ranges that are protected by Proxy Firewalls, and IP address ranges that are prohibited from being Native Hosts. There are three options available for configuring the Routing Certificates “Use Firewall”, “ Do not use Firewall”

and “No Native Associations”. The options “Use Firewall” and “ Do not use Firewall” are not security relevant, as they are used to route IP datagrams. The “No Native Associations” option is security relevant to the evaluated configuration, because if the No Native Associations Routing Option is set, the Companions behaves as if it is in Firewall Mode, even if it is in Intermediate Mode with respect to the associated IP address. The No Native Associations Routing option is only applicable if the Companion is in Intermediate Mode.

5 Architecture

5.1 System Overview

The Dragonfly Companion is a network security product that uses a Fortezza card and software to provide a security boundary between a host and an adjacent IP Dragonfly Domain and provides packet encryption, authentication and validation between Dragonfly Guards and Companions.

A Dragonfly Domain is defined as the set of hosts that can be directly accessed by a Guard or Companion (i.e. without intervening Dragonfly Units). All hosts within a Dragonfly Domain share the same security level and privileges. Dragonfly Units do not distinguish (from a security standpoint) between hosts within a Dragonfly Domain.

All packets reaching a PC with the Dragonfly Companion installed are evaluated by the Companion to determine if they should be released to the host PC (and vice versa), or discarded, and what form the released packet should take (e.g. encrypted or plain text). Dragonfly makes use of both symmetric (SKIPJACK), and asymmetric keys (Digital Signature) as well as the Secure Hash and the Fortezza Key Exchange as provided by Fortezza PC Cards.

5.2 Hardware Overview

The PC running Dragonfly Companion has the following hardware:

- External Ethernet
- PCMCIA Reader
- FORTEZZA Card

5.2.1 External Ethernet

The Companion host PC must have an ethernet network interface for the Companion’s Remote Port. Otherwise there is no need for the Companion.

5.2.2 PCMCIA Reader

The PC must have at least one available PCMCIA reader slot to allow it to access the Fortezza card. The reader is not security relevant, because it does not implement any TOE Security Functions. If

the PCMCIA Reader does not function correctly then the Companion cannot form an Association with other Dragonfly Units. In addition, the private keys and other symmetric keys used by the Companion are stored on the Fortezza card. These keys cannot be output from the Fortezza card.

5.2.3 Fortezza Card

The Fortezza Card provides all cryptographic services used by Dragonfly Units. The Fortezza Cards implement SKIPJACK symmetric encryption and decryption, Secure Hash, Digital Signature, Key Generation, and Key Exchange. The FORTEZZA cards also store keys and certificates.

5.3 Software Overview

The Dragonfly Companion is a collection of software components loaded into the Windows 95 operating system. These components are listed below:

- Man Machine Interface (companionsystray.exe)
- Virtual Device Driver (ndistrap.vxd)
- Login Facility (companionlogin.exe)

The Dragonfly Companion virtual device driver (VxD) was designed to include the following major software partitions:

- Dragonfly Initialization
- PCMCIA Card Interface
- User Interface
- Network (NDIS) interface

These sections have been described in section 2.2 of this document. In this sub-section, we describe the operation of the Dragonfly Companion.

5.3.1 The Windows 95 Operating System

The Dragonfly Companion resides within and operates on top of the Windows 95 operating system. The Dragonfly Companion VxD device driver relies on the Windows 95 operating system to provide domain separation and non-bypassability of the TSP. A brief description of the Windows 95 features are described below:

The Windows 95 operating system runs exclusively on the Intel x86 family of processors in protected mode. The Windows 95 operating system contains system code to run 16-bit and 32-bit applications. The 32-bit applications run is a protected memory space, while all 16-bit applications run in a single protected memory space. Windows 95 implements virtual memory features afforded by the Intel processor and uses a memory map to partition the virtual address space. Windows 95 has 4GB of virtual address space mapped as follows:

- 4GB – 2GB: Kernel code
- 2GB – 1.5GB: Dynamic Link Libraries (DLLs)

- 1.5GB – 4MB: 32-bit Windows Applications
- 4MB – 1MB: Generally unused
- 1MB – 0MB: MS DOS Memory

Because Windows 95 uses a virtual address space the memory architecture has these features: [WIN95]

- Executes only on the Intel Platform
- Executes only in Protected Mode
- Supports only Windows 3.x, Win32, or DOS applications
- Utilizes the flat 4GB Memory Model

The Virtual Memory Management (VMM) is a VxD that provides memory management functions for applications and the Windows 95 operating system. The VMM has these features: [WIN95]

- Protecting individual process memory space from incursions by other processes
- Efficiently sharing physical memory between program and the operating system
- Protecting the Windows 95 Kernel
- Elevating the burden of physical memory management to the operating system
- Eliminating memory fragmentation
- Extending the system's limited memory resources by swapping unused pages to the hard disk.

The Windows 95 operating system contains three networking protocols by default. These protocols are NetBEUI, IPX/SPX, and TCP/IP. These networking protocols are enabled or disabled from a GUI-based menu. The individual networking protocol options are also configured with the GUI-based menu. These settings are stored within the Windows 95 Registry (stored on the hard drive).

When the Dragonfly Companion software is installed, it makes changes to the Windows 95 Registry. These changes include breaking the binding between the Transport Driver Interface (TDI) (MS TCP/IP Protocol) and NDIS device driver. It adds a binding for the TDI to the Companion device driver and the NDIS to the Companion device driver. This is illustrated in Figure 2-1.

The Dragonfly Companion therefore relies on the Windows 95 operating system to protect the Companion VxD device driver, and the system network configurations (described above) stored on the hard drive.

The two assumptions made in DF_ST that Windows 95 is configured so that it accepts only IP datagrams and that Windows 95 cannot be attacked through network protocols below the IP layer was be verified during evaluation. The User Manual (DF_UM) was updated with procedures for disabling other networking protocols. Lower level protocol messages (i.e., below the IP layer) cannot be used to bypass the Companion as they will be filtered at the IP layer (other protocol packets were dropped and IP packets were processed with the security policy of the Companion). So the Companion cannot be bypassed by lower layer protocol messages.

5.3.2 Dragonfly Operation

The Dragonfly Companion runs as a device driver under Windows. The code runs as a background task waiting to be called to accomplish some task. There are several entry points into the code, including the following:

1. Local Port Ethernet Packet Processing,
2. Remote Port Ethernet Packet Processing,
3. Timer Interrupt,
4. User Interface Control and,
5. PCMCIA Card Insertion and Removal.

The Ethernet Packet Processing transfers packet data to and from the external network port (remote) or from the local computer's TCP/IP protocol stack (local). When packets are received on either port, they are checked to see what they are. Based on their contents and destination, the input routines can either KILL (drop) the packets, PASS them directly to the other port, or GET them into the Dragonfly code for processing. Bypassing of the Dragonfly code is done where possible to increase performance by removing redundant copying of data. The following chart shows how packets are classified. As described earlier in section 4.3, there are several operating modes for the Companion. As far as the Companion is concerned, there are three types of packets – ARP/RARP, IP and non-IP. In this case, the term "Non-IP" refers to all ethernet packets which are not IP or ARP/RARP and "Other IP" refers to all IP packets which have not been referenced in the immediately preceding lines of the table.

Operating Mode	**Packet**	**Assoc Type**	**From Network (Remote) Port**	**From Host PC (Local) Port**
BLOCK_ALL	Non-IP,	n/a	KILL	KILL
	ARP/RARP	n/a	KILL	KILL
	IP - PDD	n/a	GET	n/a
	Other IP	n/a	KILL	KILL *
PASS_ALL	Non-IP	n/a	*PASS*	*PASS*
	ARP/RARP	n/a	*PASS*	*PASS*
	IP - PDD	n/a	GET	n/a
	Other IP	n/a	*PASS*	*PASS* *
INTERMEDIATE PROTECTION MODE	Non-IP	n/a	*KILL*	*KILL*
	ARP/RARP	n/a	*PASS*	*PASS*
	IP - UDP port 15		GET	n/a
	IP - ICMP Reply to Dfly msg		KILL	n/a
	IP, Dfly PING		GET	GET *
	Other IP	NATIVE WR =	*PASS*	*PASS* *
		Other	GET	GET *

FIREWALL MODE	Non-IP	n/a	*KILL*	*KILL*
	ARP/RARP	n/a	*PASS*	*PASS*
	IP - UDP port 15		GET	n/a
	IP - ICMP Reply to Dfly msg		KILL	n/a
	Other IP		GET	GET *

Table 5-1 Dragonfly Companion Packet Input Routine Decision Table

* First check packet's source IP address to see if we can get our IP address from it.

When a GET decision has been made, the receive processing routine obtains an empty packet from the packet memory heap, copies the received packet into it, and places it on the appropriate received packet list.

The Timer Interrupt updates a global "tick" register and checks to see if the Fortezza card needs processing. First, it checks to see whether there is data from a completed operation available in the card. Then, if the Fortezza card is idle, a check is made whether there is a packet on the Fortezza input Queue that can be loaded into the Fortezza for processing.

Under Windows 95 the Companion software runs at the same level for any input – in other words, the code's various threads do not interrupt each other.

5.3.2.1 Dragonfly Data Objects

In Intermediate Protection Mode the input routines can pass IP packets by the trusted core, but only if the core has set up a NATIVE, write equal association already for that packet to use. All other IP packets that are not dropped are put on a queue to be processed by the Dragonfly's trusted core. The trusted core is responsible for making all remaining decisions on what packets are allowed to be transmitted. These decisions are based on information maintained by the trusted core in its Host and Association Tables. The significant data objects used by Dragonfly are listed below. Detailed Descriptions of the Dragonfly Data Objects and Message Formats are given in Appendixes A, B, and C.

Dragonfly Data Object	Description
Packet Object:	Trusted Memory Structure Assigned to each Datagram containing the Port associated with the Packet, the Security Level, the PSV, and a pointer to the DG_Carrier and the Packet Header.
Datagram Carrier (DG_Carrier):	Structure containing a 2K byte Datagram. Generally referred to as a "Packet"(a "Packet" contains a network packet at any protocol level from the network, IP.)
Trusted Packet List	Linked list of trusted Packets used only by the Trusted Core
Free Packet_Object List	Linked list of trusted Packet_Objects used only by the Trusted Core
Wait Queue	Linked list of Packet_Objects waiting for a future event
One-time Schedule	Table of function calls scheduled to execute at a designated time

Table	
Permanent Schedule Table	Table of function calls scheduled for repeated execution at a designate rate
Fortezza Input Queue	Linked list of Packet_Objects waiting for Fortezza Processing
Fortezza Output Queue	Pointer to the Packet_Object associated with the current Fortezza processing
Fortezza Common Memory	Mapped Fortezza memory where data and command chains are loaded
Association Table	Table of Associations indexed by end point's certificate Ids, containing security levels, association and release keys, and time out values(one Table per port)
Host Table	Table containing IP address of all known hosts with pointers to the appropriate Association Table entry. (One table per port)
Symmetric Keys	Symmetric Key structure contains a Ks Wrapped SKIPJACK key and a time to delete if inactive, and an absolute time to delete independent of activity.
Dragonfly Certificates	Variable length structures less than 2K bytes used to store security relevant information on a Fortezza card. All Certificates contain an ID, Issuer Name, Validity Period, data, and a Digital Signature.

Table 5-2 Dragonfly Companion Data Objects

5.3.2.2 Dragonfly Certificates

The Dragonfly Companion make use of nine certificates written on the Fortezza by the Local Authority.

1. **Root Authority Certificate:** trusted because it is in slot 0 of a properly prepared User Fortezza Card.
2. **Root Certificate**: trusted because it is signed by the Root Authority and the Dragonfly Unit validates it.
3. **Local Authority Certificate:** trusted because the Root signs it and the Dragonfly Unit validates that signature.
4. **User Certificate:** trusted because the Local Authority signs it and the Dragonfly Unit validates that signature. The User Fortezza Certificate includes information about the security levels and privileges of each of the devices two ports. A Companion identifies itself to other Dragonfly Units by sending a copy of its User Certificate in a message signed by its Private Key.
5. **Audit Mask Certificate:** trusted because the Local Authority signs it. Audit Certificates specify the events that are to be reported to those devices.
6. **Certificate Revocation List Certificate:** trusted because the Local Authority signs it. It contains the list of all User Certificates that have been revoked by the Local Authority.
7. **Configuration Certificate:** trusted because the Local Authority signs it. This certificate is generated by the Dragonfly Administration System, and contains all the parameters need to configure a Companion.

8. **Routing Certificate:** trusted because the Local Authority signs it. This certificate contains information about IP address ranges that are protected by Proxy Firewalls, and IP address ranges that are prohibited from being Native Hosts.
9. **Softkey Certificate:** trusted because the Software Authority of ITT Industries signs it. The Companion the checks the "Softkey" certificate to see if this is a Companion card instead of a Guard card. (Only Companion-enabled cards work with Companion software).

5.3.2.2.1 Security Relevant Fields of Companion Certificates

Certificates 4 to 8 (listed above) the User Certificate, the Audit Mask Certificate, the Certificate Revocation List Certificate, the Configuration Certificate, and the Routing certificate contain configuration information and is signed by the Local Authority. Certificate 1 to 3 (listed above) the root authority certificate, the root certificate, and the local authority certificate are used to sign and verify other certificates.

In this section, the security relevant fields of Certificates 4 to 8 are described.

User Certificate:
Certificate Type: This field has the value 0xdff0 that identifies it as a Dragonfly User Certificate.
Certificate Length: The length of the certificate.
Issuer Distinguished Name: The distinguished name of the Local Authority.
Subject Name: The distinguished name of the user.
Start Time: The start time of the certificate.
Expiration Time: The time when the certificate expires.
Certificate ID: The unique ID of the certificate.
Local Security Level: The local security level
Remote Security Level: The remote security level
Local Neighborhood ID: The ID for the local neighborhood domain.
Remote Neighborhood ID: The ID for the remote neighborhood domain.
Local Privilege Vector: The privilege vector for the local domain.
Remote Privilege Vector: The privilege vector for the remote domain.
Public Key: The Public Key of the user.
Signature: The Local Authority signature.

Audit Mask Certificate:
Certificate Type: This field has the value 0xdfad that identifies it as a Dragonfly Audit Mask Certificate.
Certificate Length: The length of the certificate.
Issuer Distinguished Name: The distinguished name of the Local Authority.
Audit Mask ID: The unique ID of the certificate.
Expiration Time: The time when the certificate expires.
Audit Mask: The Audit Mask.
Audit Mask Name: The name of the Audit Mask.
Signature: The Local Authority signature.

Certificate Revocation List Certificate:
Certificate Type: This field has the value 0xdfcc that identifies it as a Dragonfly Revocation List Certificate.
Certificate Length: The length of the certificate.
Issuer Distinguished Name: The distinguished name of the Local Authority.
Revoke List ID Number: The unique ID of the certificate.
Revoked ID Count: The number of entries.
Expiration Time: The time when the certificate expires.
Revoked Certificate ID Numbers: The ID numbers of the revoked certificates.
Signature: The Local Authority signature.

Configuration Certificate:
Certificate Type: This field has the value 0xdfc0 that identifies it as a Dragonfly Configuration Certificate.
Certificate Length: The length of the certificate.
Issuer Distinguished Name: The distinguished name of the Local Authority.
Configuration Files (Contents):

[Local Port]

Security Level: The security level of the local port.
Firewall Protection: Identifies if the Companion is configured for Firewall Protection mode.

[Remote Port]

Security Level: The security level of the local port.
Firewall Protection: Identifies if the Companion is configured for Firewall Protection mode.

[Timing]

Wait for Receipt: The time (in seconds) to wait for a receipt message from its Audit Catcher.
Wait for Association: The time (in seconds) to wait to establish an association.
Receipt Retries: The number of times to retry sending a message that requires a receipt from another Companion or Audit Catcher.
Association Time to Live: The amount of time (in minutes) that an association is allowed to exist without activity.
Association Check Period: The amount of time (in seconds) which will elapse before the Companion will check if Association Time to Live has been exceeded.
Crypto Period: The maximum amount of time (in hours) that Dragonfly Unit will be allowed to maintain an association.

[AUDIT]

Audit Catcher: The IP Address of the Audit Catcher. (Multiple Audit catchers (up to five) can be specified by repeating this entry)
Audit Catcher Required: If the Companion requires an audit catcher.
Checkin Period: The amount of time (in seconds) before the Companion sends its Checkin message to the Audit Catcher.
Enable Anticipated Messages: If the anticipated messages are enabled.

[SNIU CONFIG]

MSE Port:

SNIU Name: The name of the Companion
Allow Pass Trough: If Pass All mode is allowed for the Companion.
Allow Default Pt: If Allow user to change default is allowed for the Companion.
Authority Port:

Signature: The Local Authority signature.

Routing Certificate
Certificate Type: This field has the value 0xdffc that identifies it as a Dragonfly Routing Certificate.
Certificate Length: The length of the certificate.
Issuer Distinguished Name: The distinguished name of the Local Authority.
Routing ID Number: The unique ID of the certificate.
Number of Entries: The number of entries.
IP Address: The IP Address of the host.
IP Address Mask: The subnet mask.
Type: The Firewall type.
(Note: The italic fields can be repeated, to specify multiple entries)
Signature: The Local Authority signature.

(Note: The certificates used by the Dragonfly Guard are described in section 8.1.2 of the Guard FER)

5.3.2.3 Packet Types

Dragonfly uses four main types of packets to perform all signaling
1. Native Packet: Plain text packets as produced e.g. by normal PCs. This includes ARP/RARP, IP, UDP, and TCP formatted Packets
2. Signed Datagrams: Plain Text UDP Port 15 packets signed by the Local User's Private Key.
3. Protected User Datagrams (PUDs): UDP Port 15 packets carrying encrypted data. Both the source and destination IP addresses are those of Dragonfly Units. The data portion of the packet is an encrypted user Datagram to be delivered to a Remote host.

   Type 1 PUDs -
   Type 1 PUDs are used when only two Dragonfly Units are in the path between the communicating hosts. When a Guard receives a plain text packet (a non-dragonfly packet), a security label corresponding to the security level of the Dragonfly Unit's port on which the plain text packet was received is appended to the plain text. Message integrity is maintained by appending a checksum of the embedded packet prior to encryption. The plain text user packet plus the security label and checksum is then encrypted using the "Association Key", known only to the original sending Dragonfly Unit and the destination Dragonfly Unit. Upon decryption, this checksum is used to validate the message. Once a type 1 PUD is created, it is not decrypted until it reaches the final destination .

   Type 2 PUDs -

Type 2 PUDs are used when more than two Dragonfly units are in the path between the communicating hosts. The purpose of this format is to allow intermediate Guards to validate PUDs without knowing the key used to encrypt the data. A Type 1 PUD is converted to Type 2 when it is sent to an intermediate Guard. In this case, a second symmetric key, the "Release Key", known only to two adjacent Guards, is used to "decrypt" the type 1 PUD (since the key is not the same as the key used to encrypt, the result is not plain text). The checksum of this "decrypted" data is then appended to the Type 1 PUD. When an intermediate Guard receives a Type 2 PUD, it performs the exact same checksum calculation to validate the Packet before releasing it to an adjacent network.

4. Protected Dragonfly Datagrams (PDDs): UDP Port 15 packets carrying encrypted data. Both the source and destination IP addresses are those of Dragonfly Units. The data portion of the packet is an encrypted Datagram addressed to another Dragonfly Unit, e.g. an Audit message. Type 1 and Type 2 PDDs are generated the same as for PUDs.

5.3.2.4 Local Authority-Configured Mode Control Parameters

The Local Authority has the ability to limit the operational modes available to the user through the configuration stored on the Fortezza card when it is programmed. The configuration information is written into a Fortezza certificate and signed by the Local Authority's private key. This certificate is read and validated by the Dragonfly unit during initialization. There are three configuration options related to mode on the User Fortezza Card that can be set by the local authority on the Dragonfly Administration System:

a) Firewall Mode option,
b) Pass Through Allowed option, and
c) Allow User to Change Default option.

Firewall Mode Option:
This option only has effect on the Companion when a user is logged in to the Companion. If the local authority selects this option, the trusted human user of the Companion is not able to select Intermediate Protection Mode, but can select between Block All Mode, Pass All Mode or Firewall Protection Mode. (Pass All is not in the evaluated configuration.)

Pass Though Allowed Option:
If the local authority checks this parameter the Companion will default to Pass All mode when not logged in and the trusted human user of the Companion will be able to select Pass All Mode. If this option is not selected, the default mode when not logged in will be Block All mode. Pass All mode will not be selectable by the trusted human user, unless the Allow User to Change Default option is set. Because this option causes the Companion to default to Pass All mode when not logged in, this option is not allowed in the evaluated configuration.

Allow User to Change Default Option:
This option determines the availability of the "Pass all packets before login" menu item, which dictates the availability of "Pass All" mode when **no** user is logged into the Companion. If the "Allow User to Change Default" option is NOT checked, the "Pass all packets before login" menu

item will be disabled, and the user will not be able to determine the availability of "Pass All" when the Companion is not logged in. Not selecting the "Allow Pass-Thru Mode" and "Allow User to change default mode" options will disable "Pass All" mode at all times. This option was not allowed in the evaluated configuration.

Note: These last two parameters are stored in the registry at login time and persist after logging out from the Companion. Since they are in the Registry, they can be changed by a knowledgeable user using RegEdit. In the evaluated configuration, untrusted users are not allowed on the hosts so the users are trusted not to change the registry. In addition, there is no network interface that can be used to change the registry.

Write-ups Enabled:

A Companion configured with "Write-ups Enabled" will allow received packets to be transmitted to a host at a higher security level than that of the packet. In addition to sending the packet, if the packet is one of the Dragonfly supported "Write-up protocols", Ping, FTP, SMTP and DNS, Dragonfly will generate an "Anticipated Handshake Response" to the packet being "written-up". If a packet is later received that matches the "Anticipate Response", the anticipated message is released to the low side, completing the handshake without opening a channel for user data to leak from high to low. Only FTP and SMTP commands that request no User Data are anticipated. Companions not configured to enable write-ups will block and audit all attempted write-ups. The "Write-ups Enabled" flag provides additional protection.

Note: Attempted write-downs, unless anticipated, are always blocked and audited.

Audit Required:

The Local Authority can configure a Companion to "Require Audit Catcher". In this mode, the Companion is required to maintain an association with an Audit catcher. The Companion must periodically check in with the Audit Catcher, and if a validate receipt is not received, the Companion will stop processing user datagrams until the Guard is able to successfully check in. In between check in messages, if an audit message does not receive a response, the Companion stops processing user datagrams and attempts to re-establish contact with an Audit Catcher. If Audit is not required, the Companion still attempts to support auditing. However, if an audit catcher can not be found, audit messages are buffered until the audit buffer is filled, and then sent to the Companion's log window buffer. User datagrams continue to be processed.

No Native Association Routing Option:

"No Native Associations" option prevents Dragonfly Units from establishing Native associations. This option can be selected from the Dragonfly Administration system and the information is stored in the Routing Certificate on the Companion Fortezza Card. If the No Native Associations Routing Option is set, the Companions behaves as if it is in Firewall Mode, even if it is in Intermediate Mode with respect to the associated IP address. This option is only applicable if the Companion is in Intermediate Mode. The Routing certificate is checked before the Companion forms a native

association with the host to ensure that the IP address of the host is not associated with No Native Association option in the Routing Certificate.

5.3.2.5 IP_HOLD and AC_HOLD Modes

When the PC is first booted, the Companion software listens in on the outgoing traffic to determine its own IP address. Until the IP address is received, the Companion is in a hold state (IP_HOLD). This means that all IP traffic will be blocked in Protected or Firewall mode until the IP address is determined (though non-IP packets still go through until the IP address of the host PC is determined). If the Companion is configured to require an Audit Catcher it will also be in a hold state (AC_HOLD).

5.3.2.6 Dragonfly Association Establishment

Trust between Dragonfly Units is established on an *as-needed-basis* by exchanging Digitally Signed Messages containing their User Certificates. This exchange of messages uses the Fortezza Key Exchange Algorithm to insure that the "Association Key" is known only to the two Dragonfly Units that are at the end points of the association and not any other device on the network. The Key Exchange Algorithm is also used to establish symmetric "Release Keys" that are shared between intermediate Guards and their adjacent Dragonfly Units. A Companion can validate a signed message from another trusted Dragonfly Unit in two steps:

1. Validation of the sender's User Certificate, which is included as part of all Dragonfly signed messages. All User certificates are signed by the Local Authority's private key. Validation is done using the Secure Hash and Digital Signature algorithms with the Local Authority's public key that is available in the User certificate on every Dragonfly Unit's Fortezza card.

2. Validation of the overall message, which is signed by the sender's User Certificate's private key. The sender's private key, validated above, is used in the validation process.

Guards and Companions maintain information about other Dragonfly units and Hosts in two types of tables, the Association Table and the Host Table. Each port, Local and Remote, have their own copy of these tables. Entries in the tables are made as new Dragonfly units and hosts are discovered. An illustrative example of the discovery process is as follows. The network configuration for this example is shown in Figure 5-1.

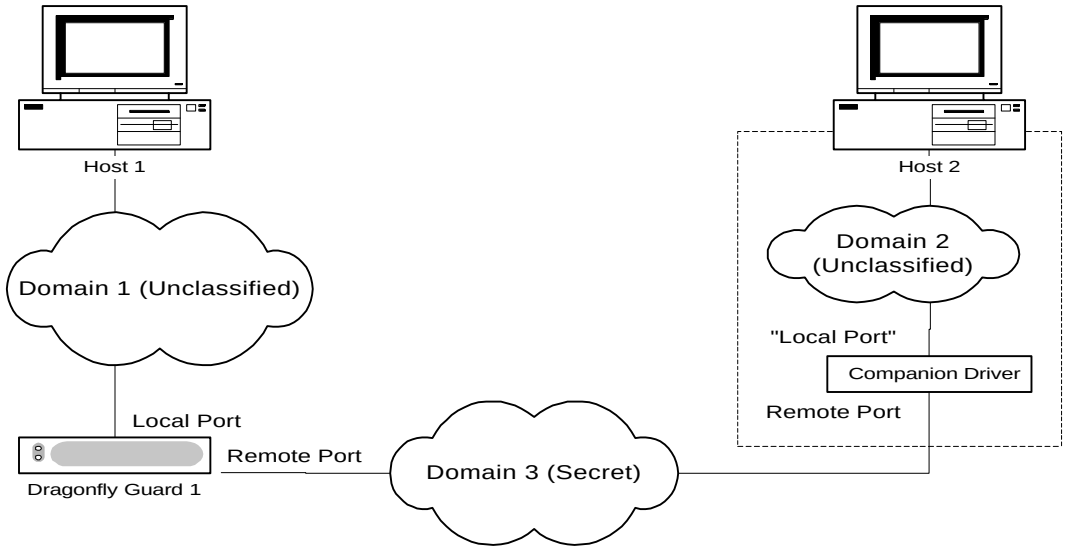


Figure 5-1: Simple Dragonfly Network with a Companion

1. Host 1 sends Packet A to Host 2.
2. Guard 1 receives Packet A and makes a "Native Host" entry for Host 1 in the local port's Host Table.
3. Guard 1 looks for Host 2 in the remote port's Host table, but Host 2 is not found.
4. Guard 1 sends a "DF Ping" to Host 2. The purpose of the "DF Ping" is to force Host 2 to respond. If Packet A is a TCP packet, the "DF Ping" takes the form of a TCP sync request. Otherwise, the DF Ping" is a real Ping (ICMP echo request).
5. Guard 1 sends an "Association Request" to Host 2. The request contains Guard 1's User Certificate (The Guard 1 user certificate contains its privilege vector).
6. Host 2's Companion receives the DF Ping and discards it
7. The Companion receives the Association Request, validates it, and saves it in the "Wait Queue"
8. The Companion, which already has a "Native Host" association for Host 2 on its local port Host Table, retrieves the corresponding Association Request from the Wait Queue.
9. The Companion checks privileges to insure the association is allowed by DAC (either in the Guard 1 privilege vector or in the Companion's privilege vector). If not, an association denied message is sent to Guard 1, and the communications are blocked. Otherwise …
10. The Companion creates a Host entry for Host 1 in the remote port's Host Table, and an Association ID entry for Guard1 & the Companion. The Fortezza Key Exchange Algorithm is used to generate a Symmetric Association key that is stored in the association table.
11. The Companion generates an "Association Grant" message. The parameters needed for Guard 1 to also generate the same key are placed in the grant along with the Companion's Certificate and sent to Guard 1.
12. Guard 1 receives and validates the Association Grant.
13. Guard 1 creates a Host entry for Host 2 in the remote port's Host Table, and an Association ID entry for Guard1& the Companion. The Fortezza Key Exchange Algorithm is used to regenerate the Symmetric Association key that is stored in the association table.

14. Guard 1 retrieves Packet A (and any subsequent packets addressed to Host 2) from the Wait Queue, extracts information from its Association Table performs the MAC checks (between the source and the destination) and if the MAC checks pass, it encrypts the packets and sends them to the Companion.
15. The Companion receives the encrypted packets, extracts information from its Association Table performs the MAC checks (between the source and the destination) and if the MAC checks pass, it decrypts and validates the message, and sends it to Host 2.

(*Note: MAC checks are not performed during association establishment. They are performed when IP datagrams are released after an association is established.*)

(Note: The validation of the Association Request (step 7) and Association Grant (step 12) messages include validating the User Certificate, checking the expiration date on the user certificate and checking the user certificate against the CRL.)

5.3.2.7 Trusted Core Main Loop Packet Processing

The main packet processing loop, repeatedly checks the following data structures for data objects that require processing:

- **Fortezza Output Queue:** If processing is complete, pass Carrier from queue to **'Extract From Fortezza'** module.
- **Fortezza Input Queue:** If the Fortezza card is idle, pass Carrier from queue to **'Fill Fortezza'** module.
- **One-time Schedule Table:** Executes any scheduled tasks that are ready to execute. E.g. re-send an unanswered association request.
- **Permanent Schedule Table:** Executes any permanent tasks for which the scheduled execution time has been reached, e.g. check-in with the audit catcher, check for expired associations, certificates, or symmetric keys.
- **Protocol Driver Input Storage:** The host PC's transport driver (on the Companion's local port). Sends packets to the Companion which are stored here to be dealt with in the timer interrupt routine.
- **Fortezza Card Status:** Check for the Fortezza Card value removed. Halt with error indication if the card has been removed.

5.3.2.8 Dragonfly Packet Processing Simplified

All software subsystems within the Trusted core communicate via a Packet_Object that identifies the packet to be processed. The following is a simplified listing of the steps involved in processing a packet from the time it is received on one Ethernet port until it is transmitted on the other port.

A packet is received by either network Interface and posted to any .

Input Packet	1. Windows calls the driver with an NDIS packet of incoming network

Processing	data. 2. The input routines make a quick decision whether to drop the packet, pass it by through an existing association, or get it for the trusted core. In the latter case, 3. A Packet Object is obtained from the "Free Packet Object List" and initialized to point to the Datagram in the DG_Carrier. The Packet object maintains all the security relevant information about the Datagram, including the port with which it is associated and the security level of the packet. The Packet Object also contains a "Process Sequence Vector" (PSV) which identifies the next processing step[s] to be performed on the packet. The ProcessInputPacket module is called to determine the steps necessary to process the packet. The processing of the packet continues until it is placed on one of three Dragonfly Queues shown below. At this point, the Packet Object's Process Sequence Vector is set to indicate the event or processing that the packet requires, the Packet Object is placed on the appropriate Queue. See Table 5-4.5. The main loop continues polling for additional data objects that require processing.
Fortezza Packet Processing	1. The timer detects that the Fortezza is idle 2. The next Packet Object on the Fortezza Input Queue is passed to 'Fill Fortezza', where the PSV is used to initiate the required Cryptographic processing. A new Packet Object and DG_Carrier are allocated to receive the results of the processing, the PSV is set, and the Packet Object placed on the Fortezza Output Queue. The original Packet Object and DG_Carrier are returned to the appropriate Free Queue. The main loop continues polling for additional data objects that require processing.
Output Packet Processing	1. The timer routine detects that the Fortezza has completed an operation. 2. The output Packet Object is retrieved from the Fortezza Output Queue and the processed data copied into the packet. 3. The DG_Carrier is added to the Output Packet Queue of the appropriate port and the Packet Object returned to the Free Queue. The main loop continues polling for additional data objects that require processing.

Required Service	**Queue Receiving the Packet Object**
Cryptographic Processing	Fortezza Input Queue
External Event e.g. Association Establishment	Wait Queue

Transmit as Native Message	Output Packet Queue

Table 5-4 Dragonfly Queues and Their Associated Services

5.3.2.9 Write-Ups

The Dragonfly Companion also allows the transfer of user data from a Domain with a low security label to a Domain with a higher security label. 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.

For Address Resolution Protocol (ARP) and Reverse Address Resolution Protocol (RARP), no MAC checks are performed and all Write Ups and responses are allowed. For FTP, Simple Mail Transfer Protocol (SMTP), Internet Control Management Protocol (ICMP), and Directory Name Service (DNS), the Dragonfly ensures that the feedback Write Down does not constitute a violation of the security policy by a patented scheme of anticipated messages.[1] Each feedback message is predicted by the Dragonfly Companion based upon the predicted protocol's command. In some cases, up to nine bytes of data from the actual feedback message is copied into the predicted message. If the actual feedback message matches the predicted message (except for the copied data), the predicted message is released. Otherwise, no message is released and there is no feedback. Responses must always be to requests for which anticipated messages have been built.

6 Documentation

This section describes the documentation provided with the Dragonfly Companion by ITT Industries. The document provided with the Dragonfly Companion is a Dragonfly Companion User Manual. The Dragonfly User Manual is addressed to the user of a single Dragonfly Companion. The Dragonfly Administration User Manual that covers the administrative system which is needed to modify security attributes is described in Annex A since the Administration System is not part of the TOE. (It does not come with each Companion.) The Dragonfly Guard User Manual (DF_GUM) covers the installation, operation, and trouble shooting of a Dragonfly Guard unit. The Dragonfly Guard is a part of the TOE, it is needed as an audit catcher for the Companion. The Audit Catcher collects audit data and updates audit masks, routing certificates, and CRLs. (The Audit Catcher does not come with the Companion.)

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

[1] The patent is number 5692124, dated November 25, 1997.

6.1 Dragonfly Companion User Manual

The Dragonfly Companion User Manual (DF_UM), provides installer guidance. A user is defined here as the person responsible for setting up a Dragonfly Companion on a host, inserting the Fortezza card in the PCMCIA reader, and maintaining the host and its interaction with the local network. This user does not have access to the Administrative System that writes Fortezza cards, and is not responsible for network administration.

The DF_UM is divided into two sections. Section I describes the installation and the basic operation of the Dragonfly Companion. Section II covers in-depth issues for the Local Authority or experienced Dragonfly software user. The issues covered in Section II are the principles behind Dragonfly security and use, the process of using a Dragonfly Guard as a designated Audit catcher to monitor network activity, configuring Dragonfly networks, glossary and licensing issues.

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 Companion.

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

	Assumption Name	**Assumption Description**
1	A.Attack_Level	Attackers are assumed to have a medium level of expertise, resources, and motivation.
2	A.Crypto_Services	Cryptographic services are provided by the User Fortezza Card.
3	A.Crypto_SOF	The cryptographic algorithms on the Fortezza card are assumed strong enough to counter at least a medium level of attack.
4	A. Local_Auth	The local authority is trusted to correctly configure User Fortezza Cards. In addition, the local authority is trusted to set the time correctly on the User Fortezza Cards
5	A.No_Lower_Level_Attack	It is assumed that Windows 95 cannot be attacked through lower level network protocols (i.e., below IP layer).
6	A.No_Other_Programs	No other programs may be installed on the host computer besides Windows 95 and the Dragonfly Companion.
7	A.No_Untrusted_Users	There are no untrusted users on the Dragonfly Companion
8	A.Only_One_IP_Port	The human user is trusted to configure Windows 95 so that there is only one network and it only accepts IP datagrams.
9	A.Physical	The Dragonfly Companion Host system is assumed to be protected from physical tampering.

10	A.User	The only user on the Dragonfly Companion is the trusted human user who has been provided with the user PIN for the User Fortezza card. The human user is assumed to be able to install the Dragonfly Companion in the evaluated configuration in accordance with the IGS Procedures. The human user is assumed able to insert the correct User Fortezza Card into the Dragonfly Companion, to connect its port to the network and to put the Companion in a proper mode. The human user is trusted not to bypass or tamper with the security enforcing functions of the Dragonfly Companion.
11	A.Windows_95	The Dragonfly Companion is installed on a Windows 95 operating system with the specified hardware configuration.

Table 6-1 Secure Usage Assumptions applicable for the User

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

Section II of the DF_UM is identical to Section II of DF_AUM. Section II provides an overview of the Dragonfly Companion, a description of using the audit catcher, a discussion of co-existence between firewalls and Dragonfly Units, and a discussion on configuring Dragonfly Networks. Automatic discovery by which Dragonfly Units 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 also covered.

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

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 the 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 the Vendor Test Suite

The evaluation team studied the tests included in DF_TPROC in detail and summarized the document as follows:

Companion Test Configuration
Section 1.1 describes the setup for the test configuration, including security levels, audit configurations, privilege vectors and enabling write-ups. The configuration of the User Fortezza Certificates, Companion configuration certificates and Audit Masks requires use of the Administration System, which is outside the TOE. It is important to note that all the Companions in the vendor test configuration are created to be their own domains (i.e. In the Administration System, the button that says "Create a domain for this Companions" should be checked as "Yes".)

Companion Access Tests
Section 1.2 gives the access control tests for *ping*s and FTP PUTS and GETS in Table 1-2. These tests check out every combination of DAC, MAC, and other configuration settings allowed by the test configuration setup.

Companion Specific Tests
Section 1.3 contains Companion-specific tests to exercise audit catcher required, Firewall Mode, and other configuration settings. This section lists test steps 1 – 45 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 (updates to the Routing Certificate were tested during independent testing), termination of associations with Companions 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 Companion, and various integrity checks. Some of the test steps require use of NetXray to capture, change, and resend messages. At the beginning of this section the rationale for all the steps is given.

Companion GUI Tests
Section 1.4 lists the GUI tests for the companion software. The tests listed include tests like removing the Fortezza card, re-starting the PC, closing windows. This section lists test steps 46 – 98 and at the beginning of this section the rationale for all the steps is provided.

Companion Options Tests
Section 1.5 lists the Option tests available for the Companion. The tests listed include testing the Companion options like "Block All Packets Before Login". This section lists test steps 99 – 110 and at the beginning of this section the rationale for all the steps is provided.

Audit Function Tests
Section 1.6 provides a table for the Audit function testing, this table identifies each required audit event as specified in the Security Target, the corresponding audit event number, the audit event name, and a description of the test used to verify the audit function. The audit events identified for testing are the audit event numbers 2-8, 10-12, 15, 17, 18, 23, 25, 29, and 33-43. (See DF_ST, Table 6.6 for a complete description of these audit event numbers.)

7.1.2 Vendor Test Configuration

The test configuration used by the vendor is described exactly by a configuration drawing (see Figure below) and Tables 1-1 and 1-2 of the DF_TPROC, which describes how to configure each of the three Dragonfly Guards and five of the Dragonfly Companions used in the test configuration. The figure below 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.

In addition to the Dragonfly software contained in the Dragonfly units, the vendor to augment testing used the following software:

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

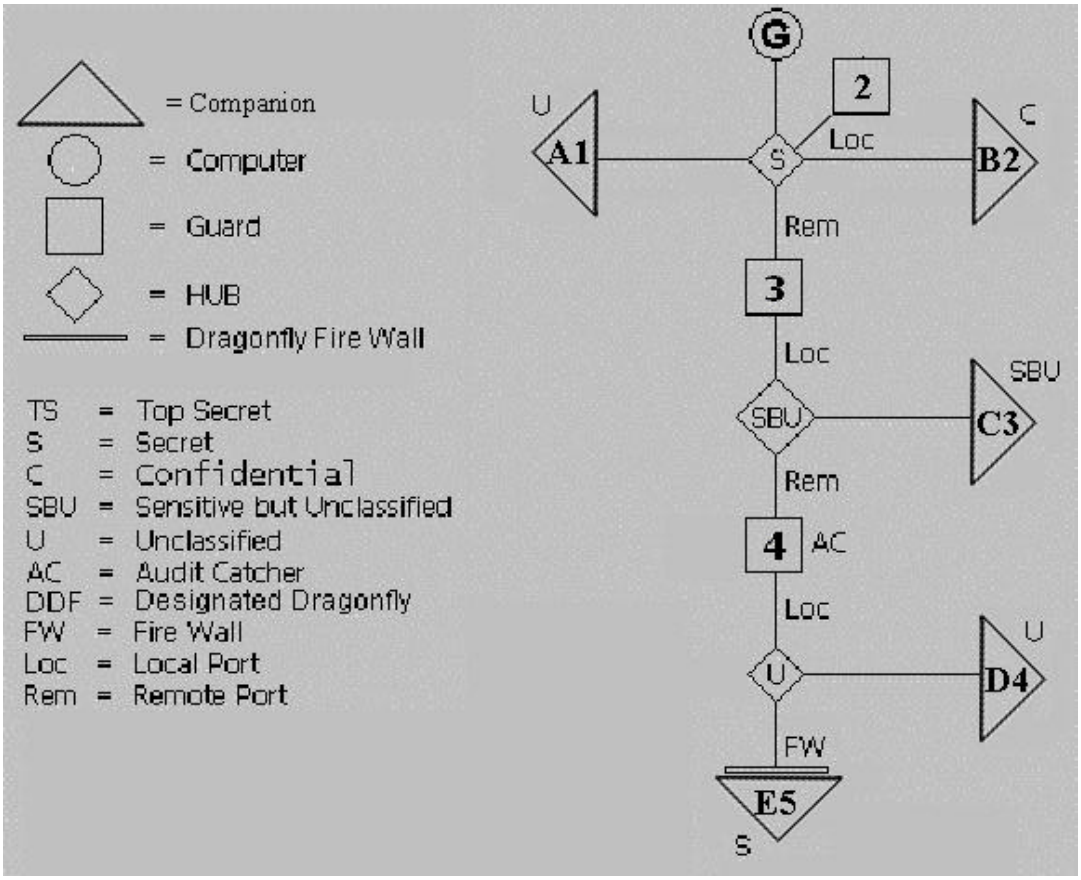


Figure 7-1 Vendor Test Configuration

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 rationale for the test procedures is also given at the beginning of each section in DF_TPROC. The evaluation team studied the test procedures in detail the tests listed in the DF_TPROC document and the mapping of the test procedures to the TOE Summary Specification in DF_ST and DF_HLD document. The evaluation team observed that some of the test

procedures specified by the vendor in DF_CD referenced a step in the test procedures without the other necessary steps to get there, so the team added the necessary steps and procedures to get to the test step referenced in DF_CD. The evaluation team came up with a list of additional independent tests required for the security functions listed in TOE Summary Specification in DF_ST. Some of the vendor test procedures were inadequate to test the security functions, the evaluation team will not be performing these tests, instead these tests were substituted with Independent Tests or other vendor test procedures that were appropriate. The security functions which the evaluation team considered were weak in the vendor test suite and decided to perform its Independent Tests are:

1. Use of Fortezza Key Exchange Algorithm
2. Domainless Companions
3. Security Levels
4. Dominance Relationship for the Security Levels
5. FTP commands
6. SMTP commands
7. ICMP Requests and Responses
8. Verifying Fortezza Card Time is set at the Local Authority Workstation
9. Verifying Companion gets its time from the Fortezza card and not the host PC.
10. Symmetric Key Expiration
11. Non-bypassability of the TSP
12. TSF Domain Separation
13. Windows 95 and the Untrusted Network Interface
14. No Native Associations Routing Option
15. Reaching Modes
16. Updating Routing Certificate

The details on the tests performed in each of these cases is described in section 7.3.2.

7.1.4 Testing Approach

The vendor uses black box testing methods to verify the claimed security functions. The testing occurs at the Dragonfly Companion interfaces. These interfaces are described in DF_HLD and DF_IFD. The test suite serves as regression tests to verify the proper operation of new versions of the Companion software. This test suite does not include any of the module level tests used during software development and integration.

The evaluation team selected the tests in the vendor test suite that mapped to the TOE Summary Specification in DF_ST. For the security functions which the evaluation team considered were weak in the vendor test suite were supplemented with Independent Tests. The evaluation team determined that this is sufficient as all the security functions listed in TOE Summary Specification are tested.

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 3.02, build 129, showing that all documented tests in the DF_TPROC had been successfully

run. The evaluators confirmed that the expected results were obtained for the vendor tests listed, corrective actions were taken when problems were found. Some problems were uncovered by the evaluation team when it re-ran vendor tests, the evaluation team reported discrepancies to the vendor. The discrepancies reported were with respect to the expected responses for the vendor tests. The evaluation team could not get the expected response for some of the vendor tests. It was apparent to the evaluation team that the discrepancies uncovered did not have a major impact on the security functionality as documented in the DF_ST and that changes have to be made to vendor documentation. The vendor reviewed these discrepancies and made the necessary changes to DF_TPROC. The evaluation team confirmed that the changes made were correct and consistent. By the end of the test period, the evaluators confirmed that the expected results matched the actual results for all the vendor test procedures and the independent tests it ran.

7.2 Modifications to the TOE Security Functions (TSF)

The vendor had to make a change to the TOE Security Functions (TSF) within the Dragonfly Companion currently under evaluation. The modified Companion will further restrict the types of data packets processed. The evaluation team proposed a two-phase process whereby this change can be accommodated with minimum impact on the evaluation.

Phase one of this process was the vendor responsibility, it proposed that the vendor perform the following:

1. Provide a description of how packets are processed before and after the change
2. Review of all evaluation documents created by vendor and update these documents if necessary.
3. Review the Test Procedures and augment procedures to include coverage of change.
4. Re-run testing as prescribed in the Test Procedures. Compare the test results with the current test procedure results and note any changes or additional tests.

Phase two of this process was the responsibility of the evaluation team.

The vendor provided the updated documents and the evaluation team reviewed them for completeness and appropriateness. The change was a minor change and impact statement provided by the vendor listed all the changes to their documentation and also described the additional tests they conducted to verify the change.

Summary of the change: Previous version of the Dragonfly Companion software dealt explicitly with packets in three categories,

a. IP packets,
b. ARP packets, and
c. RARP packets.

Packets received by the Companion software that were not identified as one of these three categories were passed by the companion to the opposite port without any checking, i.e., packets received from the Microsoft TCP/IP stack were passed to the NDIS driver, and packets received from the NDIS driver where passed to the Microsoft TCP/IP stack. This most likely did not result in a security

problem, since the TCP/IP stack is unlikely to produce non-IP, ARP, or RARP packets. Likewise, it is unlikely to accept such packets.

However, there is no need for the Companion to rely on non-dragonfly code to drop these packets. Therefore, the companion code has now been modified to explicitly discard any packets that do not fall in the above mentioned three categories.

Documents effected by the change are :

a. Dragonfly Companion Informal Correspondence Demonstration (DF_CD)
b. Companion TOE Configuration Management (DF_CM)
c. Dragonfly Companion Descriptive High Level Design Document (DF_HLD)
d. Dragonfly Companion Informal Functional Specification (DF_IFS)
e. Dragonfly Companion Security Target (DF_ST)
f. Dragonfly Test Procedures (DF_TPROC)

The evaluation team reviewed these documents and determined that the changes made to these documents were appropriate and complete.

The vendor added additional tests to their test suite to test this change, these additional tests were related to the product change. The vendor re-ran most of the tests described in DF_TPROC.

The tests that were **not** re-run by the vendor are the following:

1. Companion Specific Tests (section 1.3)
   Test Procedures 23 – 27 that check if the Companion properly handles and reports certificates with invalid or expired dates and time.
   Test Procedure 45 that verifies SMTP commands allowed for the Companion.

2. Companion Options Tests (section 1.5)
   Test Procedures 99 – 108 that test the various options available for the Companion.

These tests are not related to the changes made to the Companion Software and therefore the evaluation team feels that the vendor testing effort was sufficient after the product change.

The test results were provided to the evaluation team in hard-copy form. The evaluation team reviewed these test results and was satisfied with the testing effort of the vendor after the TSF modifications. The additional tests in the vendor test suite involved sending sample non-IP packets (IPX packets) to the Ethernet port on the Companion and verifying that these packets were dropped by the Companion. The evaluation team re-ran this test to verify the changes made to Companion Software. In addition, the evaluation team also ran additional independent tests related to the changes made which involved sending SMB/NetBIOS packets to the Companion and verifying that these packets were also dropped by the Companion. The evaluators were satisfied with the testing effort and confirmed that the changes made to the Companion Software were accurate and consistent.

7.3 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.

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 Companion initialization and all audit events for the test for each Companion involved; and NetXray snoop output when part of the test.

The DF_TPROC document identifies the goal of the test by providing a rationale for each test procedure listed in it.

The TOE test configuration used by the vendor to run their functional test suite is consistent with the evaluated configuration in the ST. This includes the environment requirements and assumptions. (ST section 1.1 TOE Identification, section 2.1 Evaluation Scope).

The vendor test procedures are at sufficient detail to enable reproducible initial conditions and test results, all of the tests reproduced by the evaluation team have demonstrated this.

7.3.1 Rerunning Vendor Tests

The evaluation team had several goals in choosing to rerun some of the vendor tests. The test configuration for re-running the vendor tests was identical to the vendor test configuration illustrated in Figure 7-1. The team reran all the test listed in Table 1-2 and Table 1-3 of DF_TPROC to show that the team's test configuration is setup and operating as expected, and that the documented expected results are obtained from the test configuration. The team also reran the vendor tests selected (vendor tests that mapped to the TOE Summary Specification) from the vendor test suite.

7.3.2 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 vendor tests provided were further explored. The test configuration usually involved two Companions and a Guard (as an audit catcher) at different levels with a native host running NetXray in between to capture and modify packets as required.

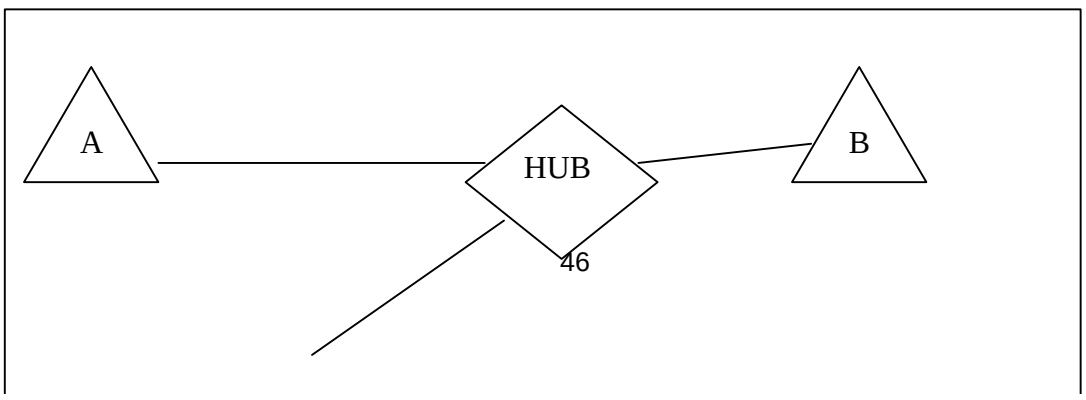


Figure 7-2 Independent Test Configuration

The independent tests conducted by the team, with vendor participation and assistance. The independent tests identify the security function being tested and the rationale for each of the tests. The expected responses for each test procedure were verified with the actual responses. The actual test results of the independent tests comprise an annotated copy of the Evaluation team's test plan, with handwritten notes as to outcome and any unexpected behavior of the test procedures.

The security functions which the evaluation team considered were weak in the vendor test suite and decided to perform its Independent Tests are:

7.3.2.1 Use of Fortezza Key Exchange Algorithm

The Companion uses the Key Exchange Algorithm (KEA) to establish an association with other Dragonfly Units. The symmetric key derived is known only to the Companion and the other Dragonfly Unit. Since, the Companion uses NSA approved Fortezza cards to perform this cryptographic function (KEA) The assumptions A.Crypto_Services and A.Crypto_SOF in DF_ST should satisfy this security function.

7.3.2.2 Domainless Companions

The local authority can configure a Companion so that its local side does not represent a unique domain. In such a case, the Companion is represented as a member of a pseudo domain. The evaluators configured domainless Companions using the Dragonfly Administration System and performed the privilege access tests for these domainless Companions.

7.3.2.3 Security Levels

Dragonfly Companions implement the following security levels:
Unclassified, Sensitive but Unclassified, Confidential, Secret and Top Secret
The security level Top Secret was not tested in Table 1-2 and Table 1-3 of DF_TPROC, so the evaluators added a test to configure the Companion at the Top Secret security level.

7.3.2.4 Dominance Relationship for the Security Levels

Dragonfly Companions implement the following dominance relationship for the security levels:
Top Secret strictly dominates Secret. Secret strictly dominates Confidential. Confidential strictly dominates Sensitive but Unclassified. Sensitive but Unclassified strictly dominates Unclassified.

The dominance relationship for security level Top Secret was not tested in Table 1-2 and Table 1-3 of DF_TPROC, so the evaluators added a test to configure the Companion at the Top Secret security level and verify that it strictly dominates the Secret security level.

7.3.2.5 FTP commands

The vendor test suite consisted of testing FTP GET and FTP PUT commands, so the evaluators added additional tests to verify that the other FTP commands do not violate the security policy of the Companion. The alternative FTP commands tested were cd, delete, dir, lcd, ls, mdelete, mdir, mget, mkdir, mls, mput, open, pwd, recv, remotehelp, rename, rmdir, and send.

7.3.2.6 SMTP commands

The evaluators added this test to try all SMTP commands to verify that the security policy of the Companion is not violated, but could not establish a Telnet session to the SMTP port of the Mail Server from the Companion to try all SMTP commands. Therefore, this test was not performed.

7.3.2.7 ICMP Requests and Responses

The evaluators added this test to verify that ICMP commands operate as defined in the security policy of the Companion. The evaluators could only test the ICMP Echo requests and ICMP Unreachable destination. The ICMP requests and responses that could not be tested are ICMP Timestamp Request, ICMP Timestamp Response, ICMP Source Quench and ICMP Time Exceeded.

7.3.2.8 Verifying Fortezza Card Time is set at the Local Authority Workstation

The evaluators performed this test by setting the system time on the Local Authority Workstation (Dragonfly Administration System) ahead of the certificate expiration time for the Companion certificates and configured the Companion. When the Companion Fortezza Card was installed in a host PC and booted up an error stating that the certificate expired was displayed in the companion log.

7.3.2.9 Verifying Companion gets its time from the Fortezza card and not the host PC.

The evaluators performed this test by setting the system time on the host PC ahead of the certificate expiration time for the Companion certificates. The Companion functioned normally and when the checkin messages were sent to the audit catcher the time on the Fortezza card was reported.

7.3.2.10 Symmetric Key Expiration

The evaluators performed this test by configuring the Companion with the parameter “Association time to live” set to 5 minutes and the parameter “Max Crypto Period” set to 1 hour. When the Companion Fortezza Card was installed in a host PC , booted up and an association was established with another Dragonfly unit. When these times expired the audit catcher reported that the audit events were generated stating that the symmetric key established was deleted.

7.3.2.11 Non-bypassability of the TSP

The evaluators performed this test by disabling the Companion software on the host PC, this allowed all the IP datagrams to be passed without any control. The Companion software was re-enabled and the IP datagrams are passed according to the security policy of the Companion software.

7.3.2.12 TSF Domain Separation

Since no untrusted human users or untrusted software is allowed on the Companion host PC, the only interface available to untrusted users is the network interface. The Dragonfly Companion processes all incoming IP datagrams. So the assumption A.No_Untrusted_Users satisfies this security requirement.

7.3.2.13 Windows 95 and the Untrusted Network Interface

The evaluators performed this test by sending IP packets and IPX packets to the Ethernet port of the host PC running Companion software. The IP packets were processed according to the security policy of the Companion. IPX packets were dropped by the Companion. Further, lower level protocol messages (i.e., below the IP layer) cannot be used to bypass the Companion as they will be filtered at the IP layer (IPX packets were dropped and IP packets were processed with the security policy of the Companion). Therefore, the Companion cannot be bypassed by lower layer protocol messages.

7.3.2.14 No Native Associations Option

The evaluators performed this test by configuring a Companion in Intermediate Protection mode and editing the routing table information on the Dragonfly Administration System to have No Native Association set for a IP Address. The native host with the IP Address in the Routing Certificate with No Native Associations set could not form an association with the Companion. Other native hosts with the IP Address not specified in the Routing Certificate could form associations with the Companion.

7.3.2.15 Reaching Modes

The evaluators performed this test by configuring a Companion in Intermediate Protection Mode. When the modes on the Companion were changed to Block All, Intermediate Protection and Firewall modes, the evaluators observed that in Block All mode the Companion did not communicate with any other hosts, in Intermediate Protection mode it communicated with other Dragonfly Units as well as native hosts and in Firewall mode the Companion communicated only with other Dragonfly Units.

7.3.2.16 Updating Routing Certificates

The Routing Certificate updates are identical to the CRL and Audit Mask updates. The evaluators created a new version of the Routing Certificate by updating the routing data on the Dragonfly Administration System. The Audit Catcher for the Companion was burnt with this new information. When the Companion sent its “Checkin Message” to the Audit Catcher the Routing Certificate was

updated. The evaluators observed the audit events corresponding to the Routing Certificate updates. (Audit Event Numbers 19 and 21 as listed in Table 6-6 of DF_ST.)

7.3.3 Penetration Testing

The evaluation team performed this testing based on the assumptions from DF_ST that the only untrusted interface to the host PC running Companion Software is the network interface. The penetration testing was built on the vulnerability analysis done by the vendor. The evaluation team agreed with the vulnerability analysis provided by the vendor (as described above in section 7.4) , therefore no attempts were to disprove the vendor rationale for the countermeasures. In a network environment the host PC running Companion software could have potentially other vulnerabilities. These vulnerabilities were not considered by the vendor, therefore the evaluation team decided to run the commercial tool, Internet Security Scanner, version 5.2, against the Dragonfly Companion. 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.

This tool does not require any special configurations. There are three default profiles “Heavy”, “Medium” and “Light” for probing. The evaluators selected the “Heavy” option. The IP addresses of the hosts to be probed is also required.

This tool was run against the “Intermediate Protection Mode” and the “Firewall Mode” of the Companion. No vulnerabilities were reported by the tool in the “Intermediate Protection Mode”. The tool also could not probe the Companion in the “Firewall Mode” since it was being run from a native host.

8 Evaluated Configuration

This section describes the evaluated configuration of the Dragonfly Companion. 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 Companions. However, a threat and risk assessment must be performed on the network in question to support secure use of the Dragonfly Companions.

8.1 Evaluated Components

The following information reflects the exact configuration of the TOE examined by the evaluation team to produce the results documented in this report. The Target of Evaluation (TOE) consists of:

- ITT Industries Dragonfly Guard Model G1.2 running Dragonfly software release 3.0, Build 980908.1509. *(Refer to DF_GFER for the evaluated configuration of the Guard)*
- Windows 95 Operating System, and
- ITT Dragonfly Companion, Version 3.02, Build 129.

Untrusted users are not allowed on the Windows 95 operating system and no other programs may be installed on the ITT Dragonfly Companion host PC. These first two assumptions are enforced procedurally and are addressed in the Administrator Guidance. The Windows 95 operating system also must be configured so that it accepts only Internet Protocol (IP) datagrams.

The following information reflects the exact configuration of the Dragonfly Companion 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_UM, as amplified by information in DF_CM. In addition to the components documented here, the user also received documentation with the evaluated configuration as defined in section 6.

Man Machine Interface	companionsystray.exe	Version 1.0, Build 7; September 15, 1998
Virtual Device Driver	ndistrap.vxd	Version 3.02, Build 129, Make 86; May 19, 1999
Login Facility	companionlogin.exe	September 15, 1998
Options Facility	companionoptions.exe	December 11, 1997

8.2 Configuration and Usage Notes

This section focuses on the Dragonfly Companion Configuration, because the configuration for the Guard is available in DF_GFER and the configuration for Windows 95 is addressed in the Administrative Guidance (DF_AUM and DF_UM).

The Configuration Certificate on the Fortezza Card contains configuration options for the evaluated configuration. The trusted human user cannot change these options on the Companion; 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 options cannot be set or modified at the Companion; configuration options must be set at the Dragonfly Administration System. The trusted human user can only change Companion modes (Block All mode, Intermediate Protection mode or Firewall mode). The Administration Guidance (DF_AUM and DF_UM) instructs the correct setting for the "required" configuration selections as described in Table 8-1.

Configuration Option	**Required Setting**
Requires Audit Catcher	This must be set to "yes" in order to guarantee that the CRLs and Audit Masks are updated from the Audit Catcher. See DF_ST security functional requirement FAU_GEN.1.1, FAU_SEL.1.1.
Allow Pass Through	To allow user to change to Pass All mode. The Pass All mode is not allowed in the evaluated configuration.
Allow User to change default	This option determines the availability of the "Pass all packets before login" menu item, which dictates the availability of "Pass All" mode when **no** user is logged into the Companion. If the "Allow User to Change Default" option is NOT checked, the "Pass all packets before login" menu item will be disabled, and the user will not be able to determine the availability of "Pass All" when the Companion is not logged in. Not selecting the "Allow Pass-Thru Mode" and "Allow User to change default mode" options will disable "Pass All" mode at all times. The Allow User to change default is not allowed in the evaluated configuration.
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 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 Privilege Vector	To enforce DAC during associations established

	through the remote port.
Allow Write Ups	To allow write ups from the low side to the high side of the Companion.
Firewall Mode	To allow the Companion only to communicate with other Dragonfly units. If this parameter is not set by the local administration system the Companion defaults to the Intermediate Protection Mode. In the Intermediate protection mode the Dragonfly Companion can communicate with other Dragonfly units and native hosts

Table 8-2. Optional Configuration Settings

8.2.2 Non-Evaluated Configuration Settings

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

Configuration Option	Comments
Point to Point Protocol (PPP) and Serial Line Interface Protocol (SLIP)	The PPP and SLIP are not included in the evaluated configuration.
TPN/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

Some of the installation problems 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 trusted human user cannot change anything on the User Fortezza Card.

8.3 Target Environment

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

Several configurations of the Dragonfly Companion have been described in section 4.

8.4 Residual Vulnerabilities

Assuming that the Dragonfly Companion is installed, configured, and operated correctly, the following vulnerabilities remain. However, these vulnerabilities are not claimed to be addressed by the TOE and are therefore listed in this section.

- **Malicious or careless administrator at Local Authority.** The Dragonfly Companion 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, then Dragonfly Companion will not operate as expected.

- **Incorrect or Compromised Administration System.** The Administration System is outside the TOE, but the Dragonfly Companion 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 Companion unit, with Fortezza Card, is captured by an adversary, they will continue to function until the certificates on the Fortezza Card are revoked.

- **Denial of Service.** The Dragonfly Companion can be subject to this attack in a number of ways like
  1. The Fortezza Card could be damaged, creating a denial of service
  2. If the Companion is configured for "Audit Catcher Required" and the Audit Catcher is not available, the Companion will stop communications, creating a denial of service.
  3. Deletion or delaying of Association Request and Association Grant messages.
  4. Destruction or modification of these Protected Dragonfly Datagrams.
  5. The IP address may be incorrectly assigned by an administrator in Windows 95 or an address assignment protocol may be spoofed to cause the Companion to assign an incorrect IP address. In these cases audit records may contain an incorrect IP address. (Note: the IP address does not provide the subject identity needed to meet the FAU_GEN.1 requirement.)

- **Identified Covert Channels.** Use of anticipated response to pass information. The vendor tests show that a small amount of data is copied from the actual response to the anticipated response, and so available to pass information..[2]

9 Results of Evaluation

The Dragonfly Companion 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 Companion meets the requirement, including the functions from the Security Target that support meeting the requirement. Section 9.2 discusses the Strength of Function requirement. Section 9.3 describes how the Companion meets the security assurance requirements.

9.1 TOE Security Functional Requirements

This section contains the security functional requirements for the TOE. All of the functional requirements have been taken from Part 2 of the Common Criteria and none of them has been refined. The functional components are listed in Table 9-1.

No.	Component	Component Name

[2] See ITT Memo 98-020, “Dragonfly Anticipated Messages”.

Class FAU: Audit		
1	FAU_GEN.1	Audit data generation
2	FAU_SEL.1	Selective audit
Class FDP: User Data Protection		
3	FDP_ACC.1	Subset access control
4	FDP_ACF.1	Security attribute based access control
5	FDP_ETC.1	Export of user data without security attributes
6	FDP_IFC.1	Subset information flow control
7	FDP_IFF.2	Hierarchical security attributes
8	FDP_ITC.1	Import of user data without security attributes
9	FDP_UCT.1	Basic data exchange confidentiality
10	FDP_UIT.1	Data exchange integrity
Class FIA: Identification and Authentication		
11	FIA_ATD.1	User attribute definition
12	FIA_UAU.2	User authentication before any action
13	FIA_UAU.6	Re-Authenticating
14	FIA_UID.2	User identification before any action
Class FMT: Security Management		
15	FMT_MOF.1	Management of Security Functions Behavior
16	FMT_MTD.1	Management of TSF Data
17	FMT_REV.1	Revocation
18	FMT_SAE.1	Time-limited authorization
19	FMT_SMR.1	Security roles
Class FPT: Protection of the TOE Security Functions		
20	FPT_ITI.1	Inter-TSF detection of modification
21	FPT_RVM.1	Non-bypassability of the TSP
22	FPT_SEP.1	TSF domain separation
23	FPT_STM.1	Reliable time stamps
24	FPT_TDC.1	Inter-TSF basic TSF data consistency
Class FTP: Trusted Path/Channels		
25	FTP_ITC.1	Inter-TSF Trusted Channel

Table 9-1 – Functional Components

The following sections contain the functional components from the Common Criteria (CC) Part 2 with the operations completed. The standard CC text is in regular font; the text inserted by the Security Target (ST) author is in italic font enclosed in brackets.

9.1.1 Class FAU: Security audit

9.1.1.1 FAU_GEN.1 Audit Data Generation

ST Requirement:

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) Received Association Exists Message,
k) Association Granted,
l) Association Requested,
m) Association Unknown,
n) Association Type Change,
o) Audit Catcher List Received,
p) Audit Mask Received,
q) Bad Message Type,
r) Opening a Write Up Session,
s) Certificate or Symmetric Key Deleted,
t) Routing Table Received,
u) Save Certificate Received,
v) Routing Table Sent,
w) Internal Error,
x) Invalid Signature, ,
x) Lost Wait Queue Msg,
y) No Receipt,
z) Revoke List Received,
aa) Attempted PUD Write Down,
bb) Received by non-Audit Catcher,
cc) Release Key Unknown,
dd) Certificate Revocation List Sent,
ee) Old CRL Version,
ff) Certificate Invalid Start,
gg) Certification Expired,
hh) Certificate Revoked,
ii) Certificate Invalid,
jj) User Logs onto Companion,
kk) Mode Change,
ll) NULL Source IP Address, and
mm) Security Level Mismatch]

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [*none].*

Dependencies: FPT_STM.1 Reliable time stamps

Applicable Features:

Every Companion can be configured by the Administration System to generate audit events, the configuration of audit events is performed via the modification of the audit mask. These events include all of the ones in the requirement , as specified by function AUDIT-6 in the Security Target. The audit events are sent to the Companion's designated Audit Catcher (a Guard , only if an Audit Catcher is identified in the configuration certificate), which is configured by the Administration System. Startup of audit functions are also shown in the audit log by the first check-in message from a Companion 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, auditing cannot be shutdown once it is started up. The Audit Catcher sends the audit log to its serial port for display. Audit records include the Companion's unique name (this constitutes the subject identity, the Companion's unique name is taken from the Companion's user certificate when an association is established with the audit catcher), the IP address, the audit code and brief description (the audit code and brief description constitute the type of event and the outcome), and the date and time. Each event code can be interpreted as a success or failure depending on its meaning as described in the User Manual.

The sequence of events that ensures that the User Name in the Audit Record is correct are:

1. The Companion Fortezza card is created on the Dragonfly Administration System. Each Dragonfly Unit has a unique user name in the User Certificate.
2. When the Companion forms an association with the its Audit Catcher. The resulting Association ID is unique. The Audit Catcher (and Companion) store, User Names, IP Address, Security Level, Association Type and Time to Live in the Association Table.
3. The Audit Event message sent to the Audit Catcher contains the Association ID. This message can only be decrypted if came from Companion associated with the Association ID due to Key Exchange during Association.
4. Audit Catcher outputs audit reports containing User Name. The User Name is looked up in Association Table based on the Association ID in the Audit Event Message.

The requirement is met by functions AUDIT-1 to AUDIT-6 in the Security Target.

9.1.1.2 FAU_SEL.1 Selective audit

ST Requirement:

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:

a) [*event type*]

b) [*none]*.

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

Applicable Features:

The Local Authority via the Administration System selects the audit events for a Companion user. 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 Companion will then include or exclude events based on the settings of the Audit Mask on the User Fortezza card. In addition, if the Standard Audit Mask is set on the Companion, the selection of events to be audited can be changed by updating the Audit Mask. The updated Audit Mask is distributed to the Companions through its Audit Catcher. The requirement is met by AUDIT-6 to AUDIT 10 and SM-3 in the Security Target.

9.1.2 Class FDP: User data protection

9.1.2.1 FDP_ACC.1 Subset access control

ST Requirement:

Hierarchical to: No other components.

FDP_ACC.1.1 The TSF shall enforce the [*discretionary access control SFP*] on [

a) *subject: source domain/Companion,*
b) *object: destination domain/Companion, and*
c) *operation: release to.*]

Dependencies: FDP_ACF.1 Security attribute based access control

Note: A Companion can be configured either as a domain or not as domain

Applicable Features:

The Companion enforces its Discretionary Access Control Policy between Dragonfly Domains. A Dragonfly Domain is a set of computers that are networked together without any intervening Dragonfly Units. The Dragonfly Companion uses bit vectors called Privilege Vectors for Discretionary Access Control (DAC) between Dragonfly Domains. Each bit represents a Dragonfly Domain. 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. DAC checks are performed at the time an Association is formed. The local authority can configure a Companion so that its local side does not represent a unique domain. In such a case, the Companion is represented as a member of the pseudo domain. Companions in pseudo domains cannot be individually specified in the privilege vectors of other Dragonfly Units.

A subject in this case is a “host in the source domain”, the object is the “host in the destination domain”, and the operation “release to” releases the IP datagrams from a host in a source domain to a host in the destination domain with respect to the DAC and MAC policy.

DAC-1 to DAC-6 are the functions described in the Security Target that meet this requirement.

9.1.2.2 FDP_ACF.1 Security attribute based access control[3]

ST Requirement:

Hierarchical to: No other components.

FDP_ACF.1.1 The TSF shall enforce the [discretionary access control SFP] to objects based on [privilege vectors and the four modes: Block All, , Intermediate Protection and Firewall Protection, and the No Native Associations Routing Option].

Applicable Features:

The Companion enforces its Discretionary Access Control Policy between Dragonfly Domains.

There are three configuration options related to mode on the User Fortezza Card that can be set by the local authority on the Dragonfly Administration System:

a) Firewall Mode option,
b) Pass Through Allowed option, and
c) Allow User to Change Default option.

The allowable modes are determined by these configuration options, the allowable modes are Block All, Pass All, Intermediate Protection and Firewall Protection. (Pass All mode is not in the evaluated configuration.)

The allowable modes are determined by the configuration options as shown in Table 4-2 of this document.

In addition, the Local Authority can prevent Dragonfly Units from establishing Native Associations with specific IP Addresses by configuring the No Native Associations Routing Option on the Dragonfly Administration System (by selecting the Edit “Routing Data” menu from the Administration System and then selecting the No Native Associations Option.)

DAC-1 through DAC-6 and IP-2 are the functions that meet this requirement.

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 the Dragonfly Companion is in Block All mode, all IP datagrams to and from the Companion are dropped.

[3] Pass All mode is not in the evaluated configuration

2) If the Dragonfly Companion is in Firewall mode, and

a) if there are two or more Dragonfly Units between the source domain/Companion and the destination domain/Companion, then

1) If the local privilege vector for the Companion or the source Domain has the bit set for the destination domain, then the IP datagram is released if the MAC check passes

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

3) Else, the IP datagram is not released.

b) or it there is only one Dragonfly Unit between the source domain/Companion and the destination domain/Companion, then the IP datagram is not released. (I.e., native mode communication is not allowed.

3) If the Dragonfly Companion is in Intermediate Protection mode,

a) and if there are two or more Dragonfly Units between the source domain/Companion and the destination domain/Companion, then

1) If the local privilege vector for the Companion or the source Domain has the bit set for the destination domain, then the IP datagram is released if the MAC check passes

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

3) Else, the IP datagram is not released.

b) or if there is only one Dragonfly Unit between the source domain/Companion and the destination domain/Companion, then

1) If the No Native Associations Routing Option is set for the associated IP address, then the IP datagram is not released;

2) Else the IP datagram is released if it passes the MAC check. (I.e., native mode communication is allowed.]

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

Applicable Features:

DAC policy can be enforced through the use of privilege vectors and modes for the Companion. Privilege vectors can prevent communication between Dragonfly Companions (or Dragonfly Units in general) independent of their security levels. Privilege vectors are stored in each Companion's User certificate and these are exchanged during the association process. The Dragonfly Unit 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 Dragonfly Unit'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 them, if one of the Companion configured in Firewall Mode, no communication can occur with native hosts.

There are three configuration options related to mode on the User Fortezza Card that can be set by the local authority on the Dragonfly Administration System:

a) Firewall Mode option,
b) Pass Through Allowed option, and
c) Allow User to Change Default option.

The allowable modes are determined by these configuration options as shown in Table 4-2. The allowable modes are Block All, Pass All, Intermediate Protection and Firewall Protection. (Pass All Mode is not allowed in the evaluated configuration.)

In addition, the Local Authority can prevent Dragonfly Units from establishing Native Associations with specific IP Addresses, by configuring the No Native Associations Routing Option on the Dragonfly Administration System (by selecting the Edit "Routing Data" menu from the Administration System and then selecting the No Native Associations Option.)

DAC-1 through DAC-6 and IP-2 are the functions that meet this requirement.

9.1.2.3 FDP_ETC.1 Export of User Data without security attributes

ST Requirement:

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 native host is at the same security level as the remote port of the Companion from which the data is exported. '

Applicable Features:

No data is released to the local and remote ports of the Dragonfly Companion in violation of mandatory access control policy. If there are no MAC violations (and Intermediate Protection Mode is enabled), only unencrypted user data are released; 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.

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

ST Requirement:

Hierarchical to: No other components.

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

a) Subjects: Dragonfly domains/Companions,

b) Information: IP datagrams,

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

Dependencies: FDP_IFF.1 Simple security attributes

Applicable Features:

MAC policy checks are made before releasing the IP Datagrams. The details are described in section 3.2 of this document. This requirement is met by functions, MAC-1 to MAC-8.

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

ST Requirement:

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/Companion,

b) Security level of the destination domain/Companion,

c) Type of protocol (i.e., ICMP, UDP, TCP, FTP, SMTP, or DNS),

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

e) *Writeups enabled,*]

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: [

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

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

a) If writeups are disabled, no IP datagrams are released.

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

1. Internet Control Message Protocol (ICMP)

Echo Requests and Time Stamp Requests are allowed.

2. *User Datagram Protocol (UDP)*

   Domain Name Server Requests with the one question flag set are allowed.

3. *Transmission Control Protocol (TCP)*

   Domain Name Server Requests with the one question flag set are allowed.

4. *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.

5. *Simple Mail Transfer Protocol (SMTP)*

   The following SMTP Commands are not allowed: EXPN, HELP, LIST, RETR, STAT, TOP, and TURN. Everything else is allowed.

6. *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.

3) If the security level of the destination domain/companion is less than the security level of the source domain/companion (writedown), the following rules apply based on the type of protocol:

a) If writeups are disabled, no IP datagrams are released.

b) If write-ups are enabled, the following rules apply:

1. *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.

2. *User Datagram Protocol (UDP)*

   Domain server responses with only one answer are allowed.

3. *Transmission Control Protocol (TCP)*

   Domain server responses with only one answer are allowed.

4. *File Transfer Protocol (FTP)*

   Predicted responses to the allowed commands that match the actual responses are allowed.

5. *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 the [*no additional mandatory access control SFP rules*].

FDP_IFF.2.4 The TSF shall provide the following [*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

Applicable Features:

The details about how this complicated requirement is met are described in section 3.2 of this document. 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 Companion 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 1.5 of the Informal Functional Specification Document. 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 to SL-3, MAC-1 to MAC-8, IP-3, and IP-6.

9.1.2.6 FDP_ITC.1 Import of User Data without Security Attributes

ST Requirement:

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 native host is in the same security level as the remote port of the Companion on which the data is imported.

Applicable Features:

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 Companion. A native packet is received by a Companion on its remote port. 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 Companion (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 hosts and other Dragonfly Units 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 local 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 remote port of the Companion on which the packet was received. This is how the two functions, IMP-1 and SL-3, meet the security requirement.

9.1.2.7 FDP_UCT.1 Basic Data Exchange Confidentiality

ST Requirement:

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.

Applicable Features:

The Companion uses its Fortezza card to encrypt and decrypt all network packets to and from another Dragonfly Unit. A symmetric key is generated during association establishment between two Dragonfly Units and is used to encrypt and decrypt. Network packets are only released in plaintext form only after the 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.

9.1.2.8 FDP_UIT.1 Data Exchange Integrity

ST Requirement:

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]

Applicable Features:

User data, sent from a Companion to another Dragonfly Unit, 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.

9.1.3 Class FIA: Identification and authentication

9.1.3.1 FIA_ATD.1 User attribute definition

ST Requirement:

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,*
e) *Routing Certificate, and*

f) Cryptographic Keys

Note: The trusted human user on the Dragonfly Companion is the human user operating the Dragonfly host who has the User Fortezza Card and PIN. When a Dragonfly Companion authenticates itself to another Dragonfly Unit, the user is represented by the User Certificate on the Dragonfly Companion's User Fortezza Card. These user attributes apply both to the trusted human user who has possession of the User Fortezza Card and the Dragonfly Companion.

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

The companion obtains its IP address from Windows 95 from an outgoing IP datagram, rather than the IP address field in the configuration certificate.

Dependencies: ITENV.1 Cryptographic Services on Fortezza Card

ITENV.3 Dragonfly Administration System for Setting User Attributes

ITENV.5 Certificates on the Fortezza Card

Applicable Features:

Security attributes are set by the Local Authority on the Administration System and written into the User Fortezza Card. Security attributes are stored in the User, Configuration, Audit, Routing, and Certificate Revocation . A User Fortezza Card must be inserted in order for a Dragonfly Companion to start up. If the User Fortezza Card is removed, the Companion goes into Block All mode (Pass All mode is not allowed in the evaluated configuration). The Fortezza card contains a User Fortezza Certificate that is used to identify the User Role. These requirements are met by the functions ATTR-1 and SM-2.

9.1.3.2 FIA_UAU.2 User authentication before any action

ST Requirement:

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
ITENV.6 Fortezza Card PINs

Note: This requirement applies both to the Dragonfly Companion authenticating itself to other Dragonfly Units and to the human user of the Companion authenticating himself or herself by entering the User PIN for the User Fortezza Card.

Applicable Features:

The local authority assigns a PIN number to each User Fortezza Card. A user must successfully login to a Fortezza Card using the correct PIN in order to use Fortezza services. The local authority must enter the correct PIN for the local authority certificate in order to login to the Administration System. The Companion is thus associated with the User role represented by the User certificate. It is the User certificate that identifies a Companion to other Dragonfly Units and is exchanged during association establishment. During association establishment process (as described in section 5.3.2.6) the user certificates are validated (including expiration date and checks against CRLs). These requirements are met by the functions ASSOC-2, IA-1, IA-2 and IA-3.

9.1.3.3 FIA_UAU.6 Re-Authenticating

ST Requirement:

Hierarchical to: No other components.

FIA_UAU.6.1 The TSF shall re-authenticate the user under the conditions [*when the User Fortezza Card is removed and re-inserted, when the User logs off and when the host is booted up*].

Dependencies: ITENV.6 Fortezza Card PINs

Applicable Features:

The local authority also assigns a PIN number to each User Fortezza Card. A user must successfully login to a Fortezza Card using the correct PIN in order to use Fortezza services. The user must login every time the host boots, when the user logs out or when the Fortezza Card is removed and reinserted. The local authority must enter the correct PIN for the local authority certificate in order to login to the Administration System. These requirements are met by the functions IA-2.

9.1.3.4 FIA_UID.2 User identification before any action

ST Requirement:

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.

Applicable Features:

The local authority assigns a PIN number to each User Fortezza Card. A user must successfully login to a Fortezza Card using the correct PIN in order to use Fortezza services. The local authority must enter the correct PIN for the local authority certificate in order to login to the Administration System. These requirements are met by the functions IA-1, IA-2 and IA-3.

9.1.4 Class FMT: Security management

9.1.4.1 FMT_MOF.1 Management of Security Functions Behavior [Trusted Human User]

ST Requirement:

Hierarchical to: No other components.

FMT_MOF.1.1 The TSF shall restrict the ability to [*determine the behavior of*] the *functions [User Data Protection by setting the mode as allowed by the configuration options*] to [*the Trusted Human User*].

Dependencies: FMT_SMR.1 Security roles

ITENV.3 Dragonfly Administration System for Setting User Attributes

Note: The options available to the user are restricted by the configuration options set by the local authority on the User Fortezza Card on the Dragonfly Administration System.

Applicable Features:

The security state menu of the Dragonfly Companion has four options: Block All Packets, Pass All Packets, Intermediate Protection, and Firewall Protection. The trusted human user can select one of these options when logged in, based on the rules depicted in Table 4-2 of this document. This requirement is satisfied by SM-2, SM –4, SM-5, SM-6 and SM –7

9.1.4.2 FMT_MTD.1 Management of TSF data

ST Requirement:

Hierarchical to: No other components.

FMT_MTD.1.1 The TSF shall restrict the ability to [*set*] the [*audit mask, routing certificate, and certificate revocation list*] to [*the local authority].*

Dependencies: FMT_SMR.1 Security roles

ITENV.4 Dragonfly Administration System for Modifying TSF Data

Applicable Features:

Initially, a Companion uses the Audit Mask, Routing Certificate, and Certificate Revocation List stored on its own User Fortezza Card. A Dragonfly Companion's Audit Mask, Routing Certificate, and Certificate Revocation List can be updated during the Check In process with its Audit Catcher. In order to do this, the Local Authority must first create a new Fortezza card for the Audit Catcher on the Administration System. When the Audit Catcher is re-initialized and receives a Check-In Message from another Dragonfly Companion, it will send it an Audit Mask Message if the Companion's Audit Mask is out of date, a Routing Certificate Message if the Companion's Routing Certificate is out of date, or a Revocation Message if the Companion's Certificate Revocation List is out of date. The audit mask for the Dragonfly Unit must be of a standard type and it must have a Audit Catcher identified in its configuration certificate in order for the audit mask to be updated from an Audit Catcher. This requirement is met by the function SM-3.

9.1.4.3 FMT_REV.1 Revocation

ST Requirement:

Hierarchical to: No other components.

FMT_REV.1.1 The TSF shall restrict the ability to revoke security attributes associated with the [*Dragonfly Companion*] within the TSC to [*the local authority*].

FMT_REV.1.2 The TSF shall enforce the rules: [*If a certificate appears on a Dragonfly Companion's Certificate Revocation List, the Dragonfly Companion will reject packets originating from a Dragonfly Companion using that Certificate].*

Note: The TSF provides the ability to revoke certificates that 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

Applicable Features:

Initially, a Companion uses the Audit Mask, Routing Certificate, and Certificate Revocation List stored on its own User Fortezza Card. When a Certificate is revoked, the local authority generates a new Certificate Revocation List (CRL) on the Administration System. When the local authority generates a User Fortezza Card on the Dragonfly Administration System, the CRL will be stored in its Certificate Revocation List Certificate. Upon initialization, the Companion uses this CRL unless or until it is updated by the audit catcher.

If the Local Authority wishes to update the CRL for a set of Dragonfly Units automatically, this can be done by generating a new User Fortezza Card with the updated CRL for the Guard serving as their Audit Catcher. The new Audit Catcher User Fortezza Card must be generated to add the new CRL, inserted in the Audit Catcher, and the Audit Catcher restarted. When Dragonfly Units check in with the Audit Catcher, the Audit Catcher sends them the new CRL, if the new CRL is more recent than the Companion's current CRL. Dragonfly Companions will then reject packets originating from Dragonfly Units using a certificate on the Certificate Revocation List. The Certificate Revocation List is checked during the process of forming an association between two Dragonfly units (section 5.2.3.6). If an association is already established and one of the Dragonfly units is on the updated CRL then the communication is terminated.

This requirement is met by the function ASSOC-2, IA-1, IA-2, IA-3, CRL–1, CRL -2, and SM-3.

9.1.4.4 FMT_SAE.1 Time-Limited Authorization

ST Requirement:

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 unit using a User Certificate*] after the expiration time for the [*user 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

Applicable Features:

User certificates contain an expiration date. This can be set to any time within one year of the user certificate start date. The default expiration date is one year from the start date. There are two expiration times associated with a symmetric key. The first is amount of time allowed for non-use. The second is the total time that the key is valid even when it is being used.

This requirement is met by the functions ATTR-2 and ATTR-3.

9.1.4.5 FMT_SMR.1 Security roles

ST Requirement:

Hierarchical to: No other components.

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

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

Dependencies: FIA_UID.1 Timing of identification

ITENV.3 Dragonfly Administration System for Setting User Attributes

ITENV.5 Certificates on the Fortezza Card

ITENV.6 Fortezza Card PINs

Note: Two roles, User and Trusted Human User, are used to distinguish between "the user represented by the User Certificate" and "the human user". User without modification is used for "the user represented by the User Certificate".

Applicable Features:

There are two roles User and the Trusted Human User. The User is represented by the User Certificate on the Companion Fortezza Card and the Trusted Human User is the one who logs on to the Companion Fortezza Card with a PIN. This requirement is met by the functions IA-1, IA-2, and SM-1.

(Note: The local authority is an assignment for several functional components, but was not included as an assignment for FMT_SMR.1. The local authority is a role on the Dragonfly Administration System and is responsible for setting user attributes and programming User Fortezza Cards. User certificates are signed by the local authority. However, the local authority cannot login as a user on the Dragonfly Companion and the local authority role cannot be associated with a user on the Companion.)

9.1.5 Class FPT: Protection of the TOE Security Functions

9.1.5.1 FPT_ITI.1 Inter-TSF detection of modification

ST Requirement:

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

Applicable Features:

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.

9.1.5.2 FPT_RVM.1 Non-bypassability of the TSP

ST Requirement:

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.

Applicable Features:

Packets are native user packets (IP datagrams), 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, and write-up enabled, before they are released by the Companion. This requirement is met by the functions, SA-1and SA-3.

9.1.5.3 FPT_SEP.1 TSF Domain Separation

ST Requirement:

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.

Dependencies: No dependencies.

Applicable Features:

The Dragonfly Companion maintains a security domain for its own execution that protects it from interference and tampering by untrusted subjects. Since no untrusted human users or untrusted software is allowed on the Companion host PC (as per the secure usage assumptions A.No_Untrusted_Users and A.No_Other_Programs in the evaluated configuration), the only interface available to untrusted users is the network interface. The Windows 95 operating system is configured to accept only IP datagrams. (Windows 95 does not come configured to accept only IP datagrams, but Companion User Manual describes how to do this and instructs the user to do this if the evaluated configuration is desired.) The Dragonfly Companion processes all incoming IP datagrams. Lower level protocol messages (i.e., below the IP layer) cannot bypass the Dragonfly Companion device driver residing in the Windows 95 operating system. This requirement is met by the function, SA-2 and SA-3.

9.1.5.4 FPT_STM.1 Reliable time stamps

ST Requirement:

Hierarchical to: No other components.

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps for its own use.

Dependencies: ITENV.8 Fortezza Card Time

Applicable Features:

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 clock and sets its own internal clock and maintains it. The Dragonfly Software frequently compares the Fortezza card’s time to its own internal clock time and updates its own internal clock. This requirement is met by the function, TIME-1 and TIME-2.

9.1.5.5 FPT_TDC.1 Inter-TSF basic TSF data consistency

ST Requirement:

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 unit) 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 Companions only interpret TSF data from other Dragonfly Units.

Applicable Features:

Dragonfly Companions only interoperate with other Dragonfly Units in terms of shared security attributes. (Native hosts do not have security attributes.) Hence, the interpretation of security levels, privilege vectors, and other security attributes is consistently interpreted between Dragonfly Units; for example, a Secret level means the same at one Companion as another. The Companion attributes configured by the local authority are stored in the certificates; the Certificate Type field identifies the type of Certificate. (For e.g. Configuration Certificate has the Certificate Type field value of 0xdfc0 and the User Certificate has the Certificate Type field value of 0xdff0.) The Dragonfly Guard and the Dragonfly Companion use the same Configuration Certificate. There are some fields in the Configuration Certificate, that are used only by the Dragonfly Guard not by the Dragonfly Companion. The fields that are used by the Guard but not by the Companion are:

[LOCAL_PORT]
ip_address=100.1.1.10
ip_mask=255.255.255.0
ip_default_gateway=127.0.0.1
sw_interrupt=0x64
rarps_allowed=false
network_type=10baseT
network_mtu=1500
arp_respond_to=0.0.0.0 0.0.0.0
[REMOTE_PORT]
ip_address=255.255.255.255
ip_mask=255.255.255.0
ip_default_gateway=127.0.0.1
sw_interrupt=0x60
rarps_allowed=false
network_type=10baseT
network_mtu=1500
[AUDIT]
this_is_an_audit_catcher=false
SNIU_CONFIG]
time_zone=8 daylight

These non-relevant fields are assigned default values by the Dragonfly Administration System when the Local Authority selects to configure the Fortezza Card as a Companion. These fields and their default values cannot be modified by the Local Authority.

There are no fields that are used by the Companion but not by the Guard.

This requirement is met by the function, CONS-1.

9.1.6 Class FTP: Trusted path/channels

9.1.6.1 FTP_ITC.1 Inter-TSF trusted channel

ST Requirement:

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. All messages before the establishment of an Association Request are signed.

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

Applicable Features:

A companions uses Fortezza services to establish a logically distinct channel with another Dragonfly Unit, known as an association. Endpoints are uniquely and unequivocally identified during association establishment with a signed User certificate representing each endpoint. Each channel between Dragonfly Units 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 to ASSOC-4.

9.2 Strength of Function Requirement

ST Requirement:

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.

Applicable Features:

The Security Target (Section 8.2.6) 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 meets the claim of SOF-medium. The only non-cryptographic, 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}.

The vendor also provided the SOF analysis in 99-003, *Dragonfly Companion 32-bit Checksum.*

Thus, all probabilistic mechanisms for providing security services required by the Security Target (Section 8.2.6) 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 3.02 (as shown by output to the audit catcher on Checkin Message) is the TOE. The final build for the evaluation, 129. The Trusted Human User can also verify the reference for the TOE from the "Help/ About" menu from the Companion GUI. The evaluators used the "Help/About" menu from the Companion GUI to verify this during testing.*
Yes	ACM_CAP.2.2D	The developer shall use a CM system. *Confirmed by reference to DF_CM. The developer provided the evaluation team with a initial DF_CM identifying all the documents and the software with version numbers and dates. During the course of the evaluation whenever updates were needed to the documents the developer provided the updated documents with updated version numbers and dates and also an updated 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_UM. The developer provided the evaluation team with the CM documentation, this is referenced as DF_CM.*
Yes	ACM_CAP.2.1C	The reference for the TOE shall be unique to each version of the TOE. *The Companion software is labeled with the version number. Each build within a release has a separate build number, which is printed out with the release number on Check in message to its audit catcher. The trusted human user can also verify this by selecting the "Help/About" menu from the Companion GUI. Both claims were confirmed during testing.*
Yes	ACM_CAP.2.2C	The TOE shall be labeled with its reference. *The Companion software is labeled with the version number. Each build within a release has a separate build number, which is printed out with the release number on Check in message to its audit catcher. The trusted human user can also verify this by selecting the "Help/About" menu from the Companion GUI. Both claims were confirmed during testing*
Yes	ACM_CAP.2.3C	The CM documentation shall include a configuration list. *The CM documentation (DF_CM) contains a configuration list uniquely identifying all the software, which comprise the TOE and the documentation describing the TOE.*
Yes	ACM_CAP.2.4C	The configuration list shall describe the configuration items that comprise the TOE. *The DF_CM describes the configuration items that comprise the TOE in enough detail to identify each component that comprise the TOE.*

Yes	ACM_CAP.2.5C	The CM documentation shall describe the method used to uniquely identify the configuration items. *DF_CM does this for the Companion software through version number and build number. The DF_CM also states that all the components of the TOE are maintained in Microsoft Visual Source Safe (VSS), which is a CM tool.*
Yes	ACM_CAP.2.6C	The CM system shall uniquely identify all configuration items. *DF_CM does this for the Companion software through version number and build number.*
Yes	ACM_CAP.2.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *The information provided meets this requirement. The details are described 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. *The Delivery Procedures were documented in the Delivery Procedures memos (99-023 and 99-024) and DF_UM.*
Yes	ADO_DEL.1.2D	The developer shall use the delivery procedures. *The Delivery Procedures were confirmed by having the developer deliver the TOE 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 delivery procedures memos (99-023 and 99-24) address the delivery procedures for the Companion and the Guard.* *For the Companion, the software (floppy disk), the User manual, the Dragonfly Administration System User Manual, and the Companion Fortezza Card are shipped by UPS in a sealed envelope.* *For the Guard, the hardware and the manuals are shipped by UPS. The Guard software is delivered on a PCMCIA card that has an integrity checksum, which is checked during initialization.* *For the Windows 95 operating system, it is recommended in the DF_UM that the Windows 95 operating system be procured through a standard authorized source such as an OEM distributor or shrink-wrapped software from an authorized dealer.* *DF_UM provides additional background information on delivery procedures.*

Yes	ADO_DEL.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *The information provided meets this requirement. The details are described 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. *The developer provided the User Manual (DF_UM), Dragonfly Administration User Manual (DF_AUM) and Dragonfly Guard User Manual (DF_GUM) for the TOE. Section 2 of DF_UM describes the secure installation, generation and start-up procedures for the Companion. The Dragonfly Administration User Manual (DF_AUM) covers the administrative system which 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 Companion.) The Dragonfly Guard User Manual(DF_GUM) covers the installation, operation, and trouble shooting of a Dragonfly Guard unit. The Dragonfly Guard is a part of the TOE, it is needed as an audit catcher for the Companion and also for updating the audit masks and CRLs. (It does not come with each Companion.)*
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_UM 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. *The information provided meets this requirement. The details are described 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. The configuration output from the Companion Menu "View/Configuration" also confirms this.*
Class Development		
4	ADV_FSP.1	Informal functional specification
Yes	ADV_FSP.1.1D	The developer shall provide a functional specification. *This is 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. *Section 1 of the DF_IFS describes security functions, while section 2 describes external interfaces. The security functions described in section 1 of the DF_IFS are consistent with the TSS in the DF_ST. Section 2 of DF_IFS describes the following: Companion Equipment and accessories, Companion programs, Network bindings, Companion Driver Interfaces, Installation, System Requirements and the Companion GUI.*
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. DF_IFS section 2 describes the external interfaces to the Companion, these external interfaces are consistent with the interfaces described DF_UM. Since, the Companion has a GUI; exceptions, error and warning messages are automatically displayed using the GUI. The exceptions, error and warning messages were found to be consistent. For e.g. when the Fortezza card was removed the Companion GUI displayed "No Fortezza Card" and when the User had logged out the Companion GUI displayed "No user logged in". In DF_UM section II 2.4 Audit codes are listed, and section 2.2 status line codes for check-ins are described.*
Yes	ADV_FSP.1.4C	The functional specification shall completely represent the TSF. *The DF_IFS completely describes the TSF. It describes the following in detail Companion Equipment and accessories, Companion programs, Network bindings, Companion Driver Interfaces, Installation, System Requirements and the Companion GUI*
Yes	ADV_FSP.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *The information provided meets this requirement. The details are described 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 Section 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. *The DF_HLD was provided by the developer, this document describes the high-level design of the TSF.*
Yes	ADV_HLD.1.1C	The presentation of the high-level design shall be informal. *The DF_HLD is descriptive, no formal language was used.*

Yes	ADV_HLD.1.2C	The high-level design shall be internally consistent. *Confirmed during the analysis for Section 9.1 above.*
Yes	ADV_HLD.1.3C	The high-level design shall describe the structure of the TSF in terms of subsystems *The TSF consists of hardware and software subsystems. Section 1 of the DF_HLD describes hardware subsystems, and section 2 of the DF_HLD describes software subsystems. The level of granularity of subsystems is sufficient to enable the understanding of the TOE.*
Yes	ADV_HLD.1.4C	The high-level design shall describe the security functionality provided by each subsystem of the TSF. *The security functionality provided by each subsystem of the TSF is described in the DF_HLD. The DF_CD document contains a mapping of the security functions to the sections described in the DF_HLD and test procedures. This was confirmed by the evaluators during preparation of Section 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_HLD. The Companion depends on the Fortezza Card for supporting protection mechanisms, including encryption services, configuration information, and user identification and authentication.* *The DF_HLD was analyzed to assure that all the functionality needed by the environment was included. First, the DF_HLD was reviewed against the requirements for the environment in the ST to ensure that they were covered. This was found to be the case. Section 2.3 of the DF_HLD describes this dependency. The Companion also depends on the Dragonfly Administration System for the correct configuration of Fortezza cards. Section 3.5.7 of the DF_HLD describes this dependency. Also, no additional environmental dependencies (which might have been overlooked by the ST writer) were found during testing.* *This requirement is targeted at the requirements on the IT Environment.*
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 identifies all interfaces to the hardware subsystems of the TSF. Section 3 of the DF_HLD identifies all interfaces to the software subsystems of the TSF.*
Yes	ADV_HLD.1.7C	The high-level design shall identify which of the interfaces to the subsystems of the TSF are externally visible. *The externally visible interfaces to the hardware subsystems are listed in section 2 of DF_HLD. The externally visible interfaces to the software subsystems are listed in section 3 of DF_HLD.*

Yes	ADV_HLD.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *The information provided meets this requirement. The details are described 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. *The DF_HLD is accurate and complete instantiation of the TOE security functional requirements. This was done during the analysis supporting Section 9.1 and other analysis in preparation of this FER. Every function described in the TOE security functional requirement was addressed in the HLD (performed in a subsystem). The interfaces to the subsystems are consistent with the functionality described in the TOE functional security requirement and the TSS.*
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. The evaluators verified DF_CD and found that all TOE functional security requirements were mapped to a HLD subsystem. The subsystem description was also reviewed to verify that the subsystem completely implemented the function. If a function was mapped to several subsystems, then together they implemented the function. The function IA-1 was mapped to the Hardware subsystem PCMCIA Reader and Fortezza Card (section 2.2 and 2.3 of the DF_HLD) and the Software subsystems PCMCIA Card Interface and User Interface (section 3.2 and 3.3 of the DF_HLD).* *The TSS was also reviewed for consistency with the FSP.*
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 Section 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. *The information provided meets this requirement. The details are described 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 trusted human user of the Companion is the administrator, and guidance addressed to the user is found in section 2 of DF_UM.*

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 Companion center around physical protection, correct installation, and audit review (Companion's local log accessible to the menu "View/Log").*
Yes	AGD_ADM.1.2C	The administrator guidance shall describe how to administer the TOE in a secure manner. *Administrative functions for the Companion center around physical protection, correct installation, and audit review (Companion's local log accessible to the menu "View/Log"). These topics are covered in section 2 of the DF_UM.*
Yes	AGD_ADM.1.3C	The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment. *There are warnings in DF_UM to disable all networking protocols except the TCP/IP protocol, not to run any other programs or allow untrusted users on the host PC running the Companion, and that auditing does not occur in Pass All mode. In addition, Section 3 of the DF_AUM and Section 3 of the DF_UM provide guidance on the setting of configuration options.*
Yes	AGD_ADM.1.4C	The administrator guidance shall describe all assumptions regarding user behaviour that are relevant to secure operation of the TOE. *There are warnings in DF_UM to disable all networking protocols except the TCP/IP protocol and not to run any other programs or allow untrusted users on the host PC running the Companion. These warnings are sufficient because the Companion processes all IP datagrams according to its security policy and since there are no other programs running on the host PC, the Companion software is not tampered.*
Yes	AGD_ADM.1.5C	The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate. *The DF_UM describes the various modes of operation for the Companion and the security relevant issues of each mode*
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. *The DF_UM describes the various modes of operation of the Companion and the security relevant issues of each mode.. The security relevant events listed in FAU_GEN are reported to the audit catcher. The administration guidance in DF_UM is consistent with other documentation provided by the vendor DF_ST, DF_HLD, DF_IFS and DF_CD.*

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. The administration guidance in DF_UM is consistent with other documentation provided by the vendor DF_ST, DF_HLD, DF_IFS and DF_CD.*
Yes	AGD_ADM.1.8C	The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator. *Section 2 of DF_UM indicates that all networking protocols other than TCP/IP be disabled for Windows 95 as Companion only processes IP datagrams. The security requirements for the IT environment in the ST were reviewed and ITENV.3, ITENV.4, ITENV.5, ITENV.6 were found to be relevant to the Administrator. These are covered in the Administrative Guidance. The Administrator and the Administrative Guidance referred to here are not in the TOE. The security requirements for the IT Environment were reviewed and none were found to be relevant to the Trusted Companion User.*
Yes	AGD_ADM.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *The information provided meets this requirement. The details are described above.*
8	AGD_USR.1	User guidance *Not Applicable to this evaluation as it is assumed that there are no untrusted users on the Companion.*
Yes	AGD_USR.1.1D	The developer shall provide user guidance. *Not Applicable*
Yes	AGD_USR.1.1C	The user guidance shall describe the functions and interfaces available to the non-administrative users of the TOE. *Not Applicable*
Yes	AGD_USR.1.2C	The user guidance shall describe the use of user-accessible security functions provided by the TOE. *Not Applicable*
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. *Not Applicable*
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. *Not Applicable*

Yes	AGD_USR.1.5C	The user guidance shall be consistent with all other documentation supplied for evaluation. *Not Applicable*
Yes	AGD_USR.1.6C	The user guidance shall describe all security requirements for the IT environment that are relevant to the user. *Not Applicable*
Yes	AGD_USR.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *Not Applicable*
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 independent tests were added where coverage was not complete. The DF_CD provides a mapping between DF_TPROC DF_ST and DF_HLD. The test procedures listed in DF_TPROC are sufficient in detail to obtain repeatability.*
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. *The rationale for each test procedure was added to DF_TPROC to accomplish this. The rationale for the test procedures is given at the beginning of each section in DF_TPROC.*

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 vendor test procedures documented in DF_TPROC and the independent test scenarios are briefly described in section 7 of this FER. The test procedures in DF_TPROC were sufficient in detail to obtain repeatability.*
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 and a rationale at the beginning of a test procedure section. The expected test results were verified by the evaluation team consistent with the design documentation and the security claims of the TOE.*
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 Companion initialization and all audit events for the test for each Dragonfly Unit 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 version 3.02build 129. The evaluators confirmed that the expected results were obtained. The expected results were consistent with the actual vendor test results.*
Yes	ATE_FUN.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *The information provided meets this requirement. The details are described above.*
11	ATE_IND.2	Independent testing – sample
Yes	ATE_IND.2.1D	The developer shall provide the TOE for testing. *The Companion software labeled version 3.02 build 129 was provided to the evaluators. The Companion user FORTEZZA cards were also provided. In addition, the Dragonfly Guard and its required accessories the ignition card and the Guard Fortezza cards were also provided. The environment of the test configuration was as specified in the ST.*
Yes	ATE_IND.2.1C	The TOE shall be suitable for testing. *The evaluators installed the Companion software on host PCs following the installation instructions provided in DF_UM. The Companion user FORTEZZA cards were configured using the Dragonfly Administration System, which is outside the TOE.*

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 Companions, six Guards and all the required accessories in order to reproduce the vendor's test configuration as closely as possible. Only five Companions and three Guards were used for testing. 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. *The information provided meets this requirement. The details are described 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 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 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 99-003. DF_VA identifies the threats to the Dragonfly Confidentiality and/or Integrity policies, these policies are dependent on the cryptographic services provided by the Fortezza Card, except for the integrity checksum. The integrity checksum uses a Cyclic Redundancy Check implemented by ITT. The SOF analysis for this mechanism is documented in Section 9.2 and 99-003, ITT Industries, Dragonfly 32-bit Checksum.*
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. *The information provided meets this requirement. The details are described 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 99-004. DF_VA and 99-004 were consistent with other documents DF_ST, DF_HLD, DF_IFS and DF_CD.*
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. 99-004 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. The Companion provides confidentiality and integrity services, and some of the vulnerabilities identified by the vendor result in denial of service threats. Therefore, the evaluators listed such threats under the Residual Vulnerabilities section of this document.*
Yes	AVA_VLA.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. *The information provided meets this requirement. The details are described 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.*

10 Evaluator Comments/Recommendations

None.

11 Annexes

Annex A: Dragonfly Administration System User Manual

The Dragonfly Administration System User Manual (DF_AUM), dated June 1998, provides guidance on using the Dragonfly Administration System. It begins by describing the Dragonfly Communication Suite (Companion, Guard and the 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 TOE.

The administrative functions and interfaces available to the administrator include management of Deployments consisting of Editing Domains, Creating Fortezza Cards for Guards and Creating Fortezza Cards for Companions. For each element of the Deployment, the security parameters under control of the administrator includes configuration settings as local privilege vectors, Audit Catcher Configured, editing of audit masks, advanced settings 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 units 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.Local_Auth	The local authority is trusted to correctly configure User Fortezza Cards. In addition, the local authority is trusted to set the time correctly on the User Fortezza Cards

The information in DF_AUM describes all assumptions regarding user behavior that are relevant to secure operation of the Dragonfly units, and describes all security-relevant events relative to administrative functions. The DF_AUM also has instructions to set the Dragonfly Units in the evaluated configuration like disabling Pass All mode for the Companions by not selecting the “Allow Pass Through” option and “Allow User to change default” option, selecting the “Requires Audit Catcher” option and selecting the Audit Mask to be “standard”. 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 Companion Documents

DF_AUM	ITT Industries, Dragonfly Administration User Manual, Version 2.05c, October 19, 1999.
DF_CD	ITT Industries, Dragonfly Companion Informal Correspondence Demonstration, Version 1.02, 25 October 1999;
DF_CM	ITT Industries, S. Meloche Memo 99-007e, October 29, 1999; Subject: Companion TOE Configuration Management;
DF_HLD	ITT Industries, *Dragonfly Companion Descriptive High Level Design Document*, Version 1.8, 28 June 1999;
DF_IFS	ITT Industries, *Dragonfly Companion Informal Functional Specification*, Version 1.1, 19 May 1999;
DF_IMSTMT	ITT Industries, Dr. E. Wrench , Impact Statement for Dragonfly Companion TOE Security Functions Change, 19 May 1999;
DF_ST	*ITT Industries Dragonfly Companion Security Target*, Version 1.5, November 6,1999;
DF_TPROC	ITT Industries, *Dragonfly Test Procedures*, Version 3.01a, 20 May 1999;
DF_UM	ITT Industries, *Dragonfly Companion User Manual*, Version 2.05d, October 25, 1999;
DF_VA	ITT Industries, *Vulnerability Analysis of the Dragonfly Companion*, Version 1.1, 23 June 1999;
99-003	ITT Industries, S. Levin Memo 99-003; March 22, 1999, Subject: Dragonfly Companion 32-bit Checksum;
99-004	ITT Industries, S. Levin Memo 99-004, March 22, 1999; Subject: Dragonfly Anticipated Messages;
99-023	ITT Industries, S. Levin Memo 99-023b, October 12, 1999; Subject: Dragonfly Companion Shipping Procedures;
99-024	ITT Industries, S. Levin Memo 99-024, September 21, 1999; Subject: Guard Delivery Procedures;

14.2 Dragonfly Guard Documents

DF_GCM	S. Levin Memo 98-016f; October 22, 1998 Subject: Guard TOE Configuration Management;
DF_GFER	*ITT Industries Dragonfly Guard Final Evaluation Report,* Version 1.1, 29 October 1998.
DF_GST	*ITT Industries Dragonfly Guard Security Target*, Version 2.0, 29 October 1998.

14.3 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.