FinFlyISP on the CloudShield CS-2000

Implementation and handover documentation

Technical and handover information

Document version 2.0

Contents

This document contains the following sections:

Limitations
Technical limitations and information relevant to the RAVE implementation of FinFly

Configuration
Variables and other settings which influence the behaviour of the application

Compilation
How to recompile the application, and the structure of the source code

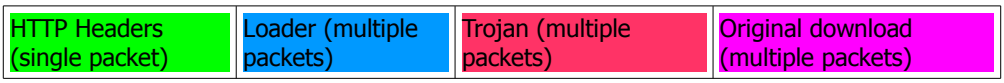
Deployment
How to deploy the application on DPPM-510 and dual DPPM-800 CS2000 systems

Limitations

Packet loss events. When transmitting packets to the client, the application is capable of detecting and recovering from certain error conditions involving packet loss. However, there are some error conditions from which it cannot recover. The limitations for recovery in binary and update mode are detailed below.

Binary Mode

In binary mode, data being sent to the client consists of the following components:



The application will detect the loss of packets sent to the client and retransmit them, with the following exceptions. Loss of any of these will cause the infection to fail.

The HTTP Header packet
The last packet of the Loader
The last packet of the Trojan

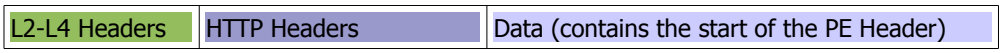
During the session, the client will be sending regular acknowledgement packets to the server. If any of these is dropped by the network, then the session may fail, as there is no event for the application to detect. This is a rare event, and in some cases will not cause a problem as the application may already be sending a second packet which will trigger a new acknowledgement in many cases.

Update Mode

In update mode, emulators are sent in response to GET requests form the client. If raw-bin is sent, then this is automatically followed by the Update Trojan configured for the target. The application will retransmit packets lost with the following exceptions:

The last packet of the Trojan

PE Detection. In order to detect a binary download on the wire it is necessary for the first packet of the download from the server to have the following structure:



Depending on the length of the HTTP Headers, there may or may not be enough of the PE Header to complete the required checks on it. Some webservers place the HTTP headers in a separate packet from the Data. In this case the application will not detect the download. If there is not enough data to confirm the presence of a PE Header of the correct type, then the detection algorithm will abort part way through, and infection will not take place. In this case, the variable `G32_PE-Detections_Aborted` will be incremented.

Internal error checking. In order to maintain as much speed as possible, the application only performs limited error checking on data it is processing. This means that it may behave in an unpredictable manner if it is given invalid data. An example of this might be a target with a trojan ID assigned which is outside the range of trojans present in the fileMatrices. It is therefore up to the administration GUI to ensure the consistency of the internal data structures that make up the configuration.

Throughput. On a dual DPPM-800 system the TCS rule files around the application can receive 10Gb/s full duplex traffic. HTTP packets are passed to the application, which has a limit of approximately 8Gb/s. Therefore, CS2000 can process a full 20Gb/s of traffic (10Gb/s in each direction) of which approximately 40% is HTTP traffic before any packets are dropped. In practical terms this should exceed the network performance in a live deployment.

Chunked HTTP connections. The application does not support chunked connections. They will work correctly from a network point of view (it does not block or drop them), but no infection will take place.

Global memory used for file storage. Due to hardware design it is required that the application stores the Loader and Trojan matrixes in a global memory area. Certain operations may be performed on this global memory area (resource algorithm modifying the loader for example). There may be possible situations where two events happening simultaneously can cause a race condition which results in data corruption. This requires two or more simultaneous infections where multiple PE headers from targets are received by the CS2000 within a few millionths of a second. If this situation ever occurs (it is not expected to) then one infection will fail but the application will recover immediately and normal operation will resume, so subsequent infections will be performed correctly.

Zero windows. The application uses TCP zero windows to hold a download server back while injection is taking place in binary mode. Some servers are more patient than others, and although most will wait for the injection to complete, this will depend on the length of time it takes to serve the injected code and the configuration of the server itself. In response to a TCP zero window the server will send TCP Zero Window probes to discover if the client is ready to receive data or not. Currently the application will act in an RFC compliant manner and respond to each of these probes with another zero window acknowledgement. The delay between each probe from the server increases, so by the time infection is complete and we are ready to resume the original download, there may be a delay before the server resumes. This delay varies from server to server and may take 30 seconds or more in some cases.

Maximum number of trojans. The application supports up to 254 trojans and 254 update responses.

Maximum number of Radius Usernames. The application supports up to 254 Radius usernames.

Configuration

The RAVE implementation of the FinFly application uses the following variables to modify its behaviour. The default values of these controls are set at compile time but may be changed via JSON commands during runtime. The default values indicated below reflect the settings present in the code at delivery. Changing these values in the source code will change the default value.

Variable	Default	Meaning
G32_FileMatrixSwitch	0	If set to 0, then payloads are served from fileMatrixA. If set to 1, then payloads are served from fileMatrixB. No other value is valid.
G32_ResourcePointerSwitch	1	If set to 0, then the algorithm which changes the icons in binary mode injection is not applied. If set to a non-zero value, then the algorithm is applied.
G32_InfectionsBeforeIgnore	65000	The number of infections to apply to a target before the target is ignored. Each target in the target table has a counter. Resetting the counter associated with the target will cause infections to be applied again.
G32_APP_VERSION_ID	100	This value does not affect the operation of the application in any way. It is recommended that this value is incremented every time a change is made to the source code. This value can be viewed in the CloudShield WMI, thus enabling the build number of the application to be determined at runtime.

In addition to the above variables, the following values are defined at compile time (using the '.define' keyword). These values are fixed and cannot be modified at runtime. If different values are required, they can be changed in the source code (the section containing them is clearly marked at the start of the file 'finfly.csm') and the application recompiled. All values are in decimal.

Definition	Default	Meaning
DEAULTSPEED	1000000	The number of clock ticks between each packet sent by the DPPM to the client during infection. Note that this value only applies to packets generated by the DPPM. A value of 1 million results in a data transfer speed of approximately 100kb/s
FILEROWS	10240	The number of rows in the file matrices used to store the loader and payloads (fileMatrixA and fileMatrixB). Each row is 1k in length, so a value of 1024 = 1Mb of file storage, 10240 = 10Mb, 102400 = 100Mb.
DPPM_800	Enabled	There is no value required. If this definition is not commented out, then the application will be compiled for DPPM-800 systems. Comment it out to compile for a DPPM-510 system. DPPM-600 systems are not supported at this time.
CL_OFFSET	64	This must be set to the offset into the loader where the value following content-length field ends.
EMULATORROWS	20480	The number of rows in the response matrix used to store the responses sent to a client in update mode. Each row is 256 bytes in length, so a value of 20480 = 5Mb of file storage.
MAXSESSIONS	10000	The number of entries in the session table. The session table only contains sessions belonging to configured targets. Any sessions from targets (ie: sessions which cannot be added to this table because it is full) will be unmodified and uninfected.
SESSION_MATRIX_SIZE	10001	This **must** be set to MAXSESSIONS + 1 for internal technical reasons.
UPDATE_MAXURLS	64	The maximum number of Hosts+URLs that can be specified for processing in Update mode
UPDATE_MAXRULES	64	The maximum number of entries in the rules table which defines the responses to send for any given URL in Update mode

Compilation

The application is designed to run on CPOS 3.0.1 or higher, and requires an IDE version 2.x to compile. It was developed using IDE 2.0.1, build number 576. The project workspace has the following structure:

Directories	
Name	Contents
DBTables	The default data that is loaded into the Target Table and the Rules Table.
Doc	Documentation in the form of the project document and this document.
MatrixTestData	Data imported into the file and response matrices at compile time.
PCAP	Wireshark PCAP files used for testing during development.
SearchTables	Data used by the regex tables within the application. Of the files in this directory the only one that should ever be changed is *URLTestData.csd* which contains the default entries in the HURL table.

Files	
Name	Contents
`FinFlyISP.csm`	The entry point for the application, preliminary filters and handlers. Defines the search tables, matrices and databases. Provides debugging logic for developmental use. At compile time, includes the files shown below. Implements some general and high level state handling.
`binary_control_module.csi`	Code to handle the TCP states associated with binary injection
`binary_module.csi`	Code to serve the actual data from the file matrices
`update_control_module.csi`	Code to handle the TCP states associated with update mode
`update_module.csi`	Code to serve the actual data from the response matrix
`resource_module.csi`	Code to implement the resource modification algorithm
`radius_module.csi`	Code to handle radius packets
`FinFlyISP.orc`	The last compiled version of the application
`FinFlyISP.adp`	The last packaged deployable version of the application

The configurable values listed in the Configuration section of this document can be found at the top of the `FinFlyISP.csm` file. In order to compile the application, select `FinFlyISP.csm` in the tree view on the left hand side of the main IDE window and press F9 or right click it and select 'Compile' from the menu. After compilation is complete, the compiler will produce a file called `FinFlyISP.orc`.

Deployment on DPPM-510 (Gigabit Ethernet) systems

In order to deploy the application onto a CS-2000, the orc file produced by the compiler needs to be packaged in an ADP (Application Deployment Package) file. Double click the `FinFlyISP.adp` file in the tree view to open the graphical interface. Within this view you will see a picture of a CS-2000 at the top, and below it, connected via some virtual cables, a black box labelled 'FinFlyISP.orc'.

Click on the FinFlyISP.orc box in the display, and press 'Delete' on the keyboard to remove it. It will disappear, along with all the virtual cables. Now drag and drop the `FinFlyISP.orc` file from the tree view on the left into the area below the CS-2000 image. Right click on the new image labelled 'FinFlyISP.orc' and select 'Create default connections...' from the context menu. This will recreate the virtual cables. Now save the ADP file using the toolbar icon or the File menu.

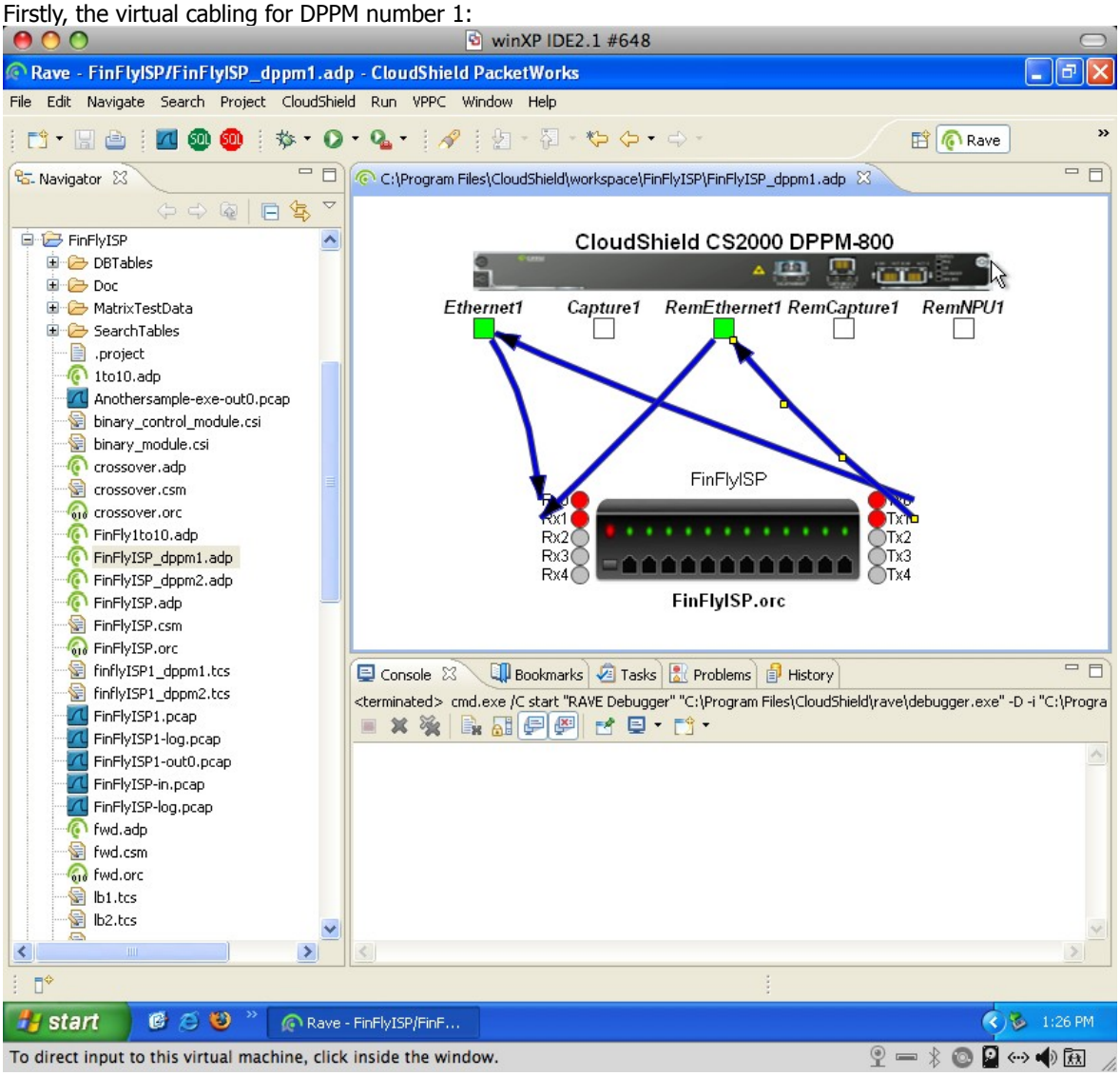
At this point the ADP file is ready to deploy on the CS-2000 hardware via the WMI.

Deployment on Dual DPPM-800 (10-Gig Ethernet) systems

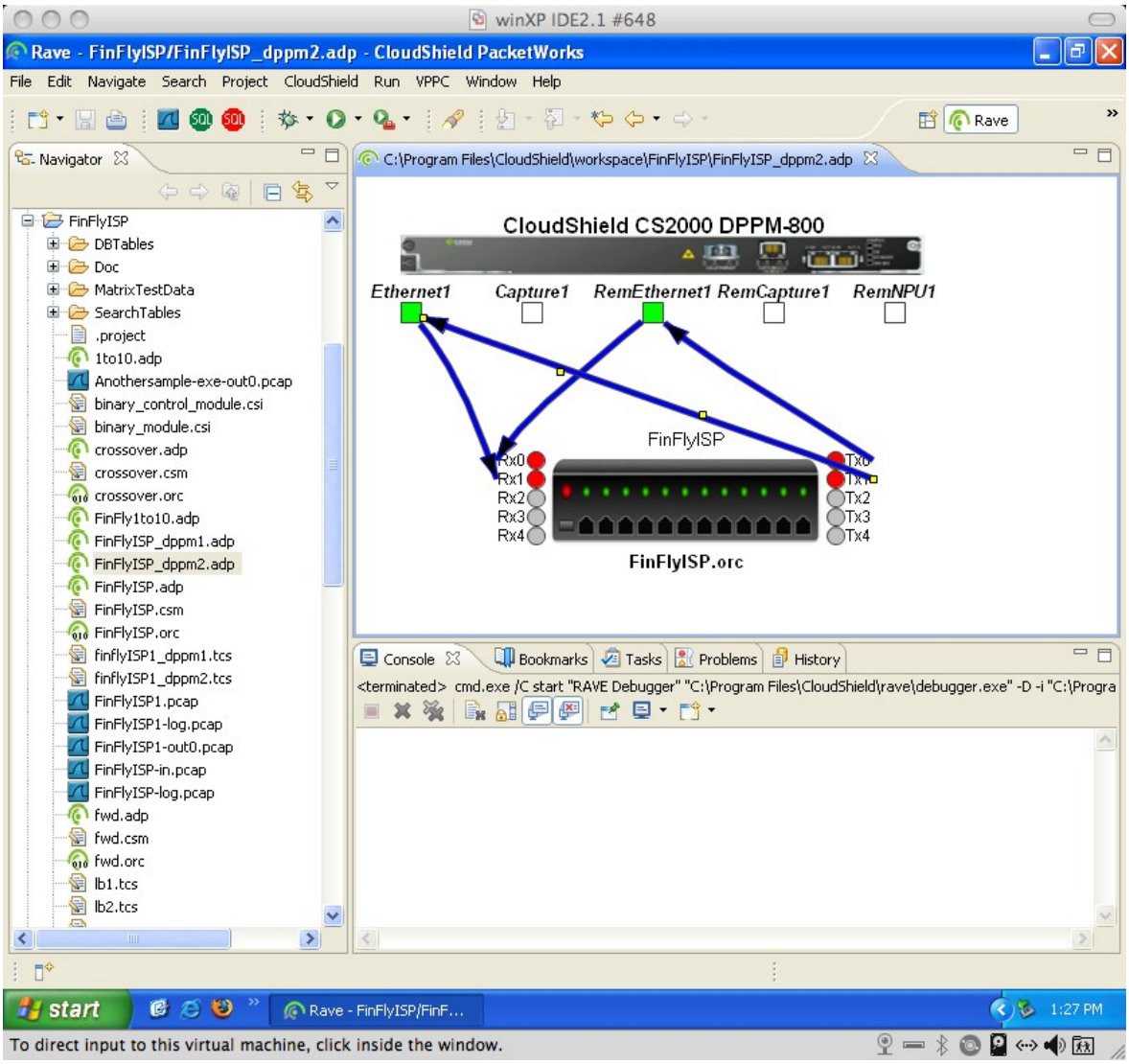
In order to run on a dual DPPM-800 system, the application needs to be installed on both DPPMs. Each DPPM runs an ADP which contains the same 'orc' file as the DPPM-510 version, but the virtual cabling in the ADP is different for each of them. Inside the project workspace in the IDE, there are two ADP files called '`FinFlyISP-10G-DPPM1.adp`' and '`FinFlyISP-10G-DPPM2.adp`'. Both of these need to be edited to include the latest version of the FinFlyISP.orc file.

For each of the ADP files, double click it to open the editor and create the virtual cables as shown in the following diagrams:

Firstly, the virtual cabling for DPPM number 1:



Secondly, the virtual cabling for DPPM number 2:



In addition to deploying the ADP files on the dual DPPM-800 system, there are two TCS configuration files that need to be deployed. In the project workspace these files are present and can be clearly identified as they have a .TCS file extension. This files should be used 'as is' and not modified.

The contents of the files are as follows:

TCS file for DPPM number 1:

```
HASH: SIP, DIP, PROTO, SP, DP, IPSWAP, PORTSWAP
RULE0: ACCEPT  PROTO=TCP, DP=1, L3TYPE=IPV4 OUTPUT=LB0
RULE1: ACCEPT  PROTO=TCP, DP=80, L3TYPE=IPV4 OUTPUT=LB0
RULE2: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE3: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE4: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE5: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE6: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE7: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE-DEFAULT: ACCEPT OUTPUT=RP0
LB0: RNP, LNP
```

TCS file for DPPM number 2:

```
HASH: SIP, DIP, PROTO, SP, DP
RULE0: ACCEPT  PROTO=TCP, DP=1, L3TYPE=IPV4 OUTPUT=LB0
RULE1: ACCEPT  PROTO=TCP, SP=80, L3TYPE=IPV4 OUTPUT=LB0
RULE2: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE3: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE4: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE5: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE6: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE7: ACCEPT PROTO=UDP, DP=1813, L3TYPE=IPV4 OUTPUT=LB0
RULE-DEFAULT: ACCEPT OUTPUT=RP0
LB0: LNP, RNP
```

These files are used to fast-track non-HTTP traffic around the application logic within the CS2000. If they are not deployed, then the application will not function correctly, as when compiled for DPPM-800 systems the application does not check to see if the traffic it is processing is HTTP, as it expects the TCS files to do that job for it.