Contents

[Quick guide to Check Point Express for F#! 2](#_Toc230083747)

[The Check Point Express Tool – milestones. 3](#_Toc230083748)

[Check Point Express Tool – milestone policy completion 4](#_Toc230083749)

[Policies: Geopolitical 5](#_Toc230083750)

[Policy: Geopolitical Risk Assessment. 5](#_Toc230083751)

[Policy: Severity 1 Geops bugs. 5](#_Toc230083752)

[Policy: Non-text element review. 5](#_Toc230083753)

[Policy: Text Element Review. 5](#_Toc230083754)

[Policies: IP 6](#_Toc230083755)

[Java 6](#_Toc230083756)

[GDI+ File Migration 6](#_Toc230083757)

[License Terms 6](#_Toc230083758)

[Policies: Interoperability 7](#_Toc230083759)

[Windows API Usage 7](#_Toc230083760)

[Launched or embedded Windows 7](#_Toc230083761)

[Protocol Documentation 7](#_Toc230083762)

[Protocol Inventory 7](#_Toc230083763)

[Policies: Privacy 8](#_Toc230083764)

[Privacy Statement 8](#_Toc230083765)

[Consent for data transfers 8](#_Toc230083766)

[Policies: Security 9](#_Toc230083767)

[Export Controls 9](#_Toc230083768)

[SDL 9](#_Toc230083769)

[Policy: Software integrity 10](#_Toc230083770)

[Virus Scanning 10](#_Toc230083771)

[Code-signing 10](#_Toc230083772)

[Doing a signed build 11](#_Toc230083773)

[Smart Card sent over Remote desktop. 12](#_Toc230083774)

# Quick guide to Check Point Express for F#!

A CPX release ID is a pre-requisite for a public release. The Check Point Express process provides a CPX release ID once a check-list of questions has been answered OK.

Reference: <http://compliance/>

Tools you need to install:

* [Check Point Express Tool](http://compliance/cpx/pages/cpx.aspx) – the process admin tool. Please install on your machine!
* [APIScan tool](http://compliance/interoperability/Pages/APIscan.aspx) – checks only MSDN public APIs are called. Please install.

Tools that are in SD and run by “make cpx”.

* [Policheck Tool](http://compliance/geopolitical/Pages/PoliCheck.aspx) – scans text for bad phrases. This is in SD under ./tools
* [Binsearch Tool](http://compliance/intellectualproperty/Pages/BinSearch.aspx) – checks for a GDI+ library. This is in SD under ./tools

Intranet services:

* [Viewpoint](http://viewpoint/) –asks if images are OK? Web submission (only needed for new images).
* [SWITrack](http://compliance/security/Pages/SWITrack.aspx) – Secure Windows Initiative projects track the SDL process.
* [Codesign](https://codesign.gtm.microsoft.com/) – web approval of codesign jobs (submission is automated).
* [Virus Checking Service](http://vcs/) – submit a share for virus scan in Dublin and Redmond.

The build steps required:

* Do signed build (the codesign submissions need a smartcard approval by the user).

e.g. users\matteot\sync\_build\_publish\_doit.cmd “CODESIGN=yes”

* Run scans, e.g. policheck, binsearch, image collection, unpack for APIscan and virus scan.

e.g. make “cpx”

The admin steps in overview:

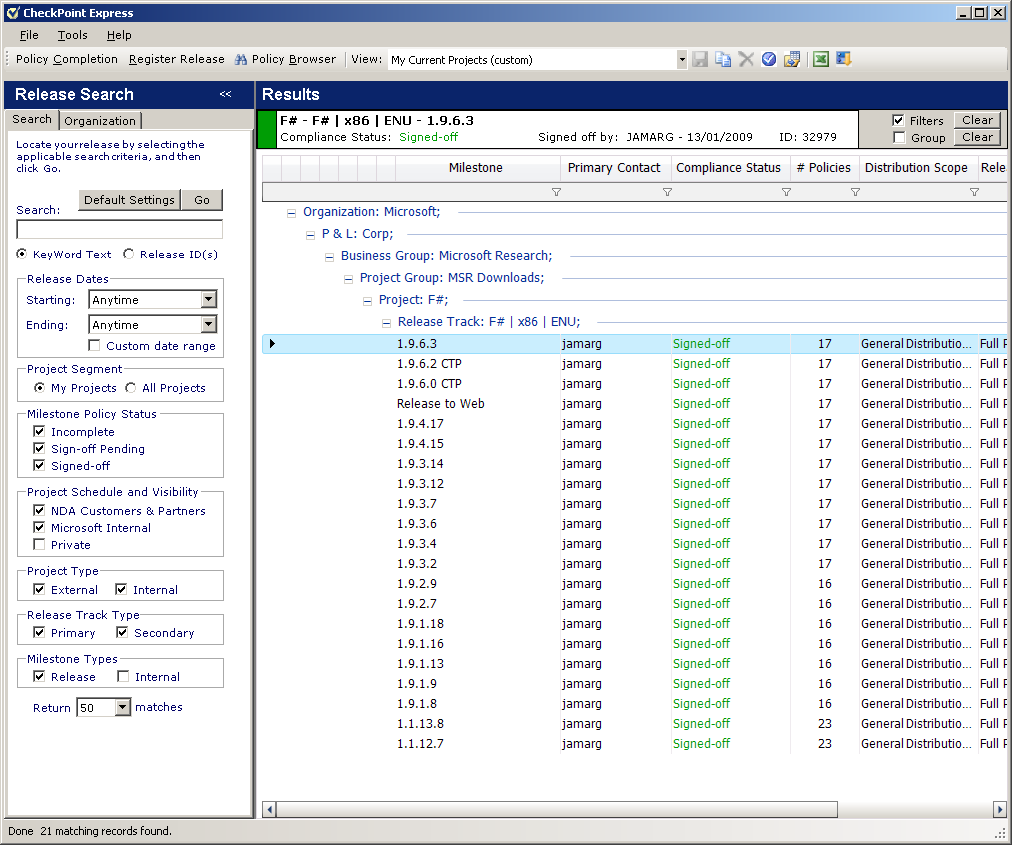
* Add a milestone for the release under the F# release track in Check Point Express tool.
* Confirm geopolitical policies OK. Check: policheck output and images.
* Confirm IP policies OK. Check: binsearch output.
* Confirm Interoperability policies OK: Run and check APISCAN output.
* Confirm Privacy policies OK.
* Confirm Security policies OK. Get switrack project id
* Confirm Software integrity OK. Check: files were signed. Check virus scan.
* Sign-off get the CPX release ID.

The following sections have guidance for completing the above.

## The Check Point Express Tool – milestones.

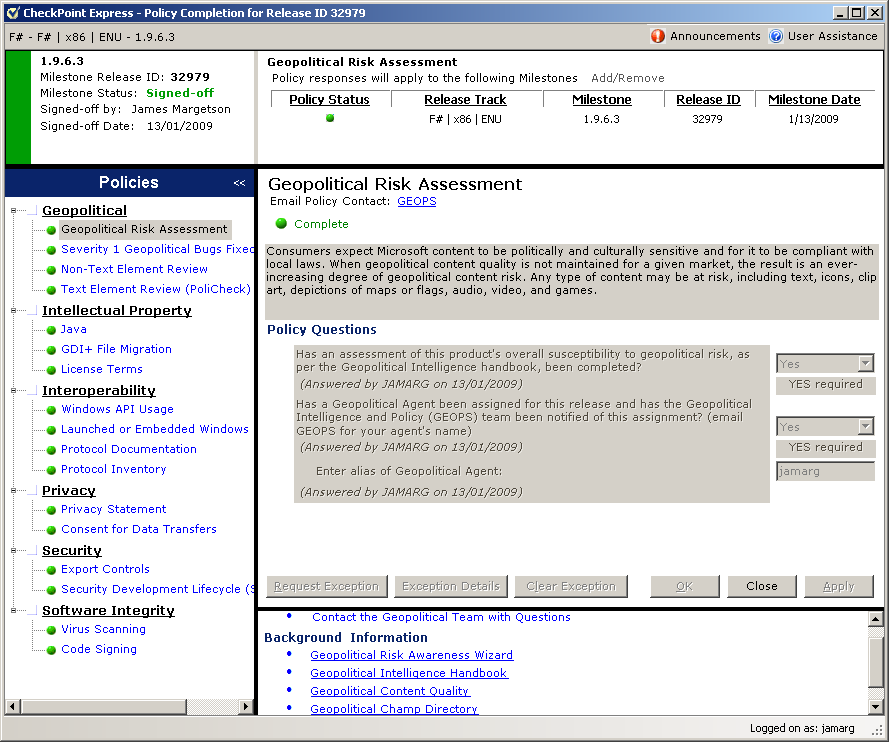
Install the Check Point Express tool from <http://compliance/cpx/Pages/cpx.aspx>

There is an “F# project” which has one track, the track has multiple milestones. Search “ALL PROJECTS” and “ANYTIME” for “F#”. For a new release, create a new milestone under the “F# release track”. I have been naming these milestones with their version number (maybe non-standard).



## Check Point Express Tool – milestone policy completion

Each milestone has checklist questions for each policy. When all are green it can be signed-off. The following sections have notes and help for completing each policy.



## Policies: Geopolitical

### Policy: Geopolitical Risk Assessment.

CPX asks if a “Risk Assessment” for “F#” has been done. It has. The geopolitical agent is jamarg.

### Policy: Severity 1 Geops bugs.

Where there any Severity 1 (no ship) geopolitical bugs for this release? If so, confirm they are fixed! These could be valid Policheck Severity 1 text issues or a media issue (perhaps a photo needed changing).

### Policy: Non-text element review.

“make cpx” collects images from the source tree into a directory, e.g. here:

FSharp-1.9.6.16-CPx\Retail\images

Check this directory. It should contain a few icons and currently no photos. Any changes to shipped images (or other media) need to be reviewed (e.g. Viewpoint) before shipping.

### Policy: Text Element Review.

“make cpx” runs the policheck tool dropping a result xls file, e.g. here:

FSharp-1.9.6.16-CPx\Retail\policheck-unpack.xls

Policheck scans for potentially sensitive words and phrases. Check the result xls file. The severity 1 results are the key ones. “Sort” the data by “Sev” then “Term” to group the Sev1 terms together.

Check:

1. The “Replacement Guidelines” column links to the guidance: <http://gpweb/Compliance/PoliCheck/TermInfo.aspx?LCID=9&TermID=79401>

For each context this gives the severity and action/recommendation.

For example, Red as a colour is OK, but Red for Communists is “Must Fix” change.

<http://gpweb/Compliance/PoliCheck/TermInfo.aspx?LCID=9&TermID=80143>

1. Binary files (e.g. pdbs) are in the unpack area (for APISCAN) and can show up false positives.

## Policies: IP

### Java

We do not ship the MS Java VM, so OK here.

### GDI+ File Migration

“make cpx” runs the binseach tool. Verify “No matrices were found” in the result file:

\\root \FSharp-1.9.6.16-CPx\Retail\done-binsearch.txt

REASON: “For legal and security reasons, there are obsolete versions of GDI+ files which should not be released with any Microsoft software.”

### License Terms

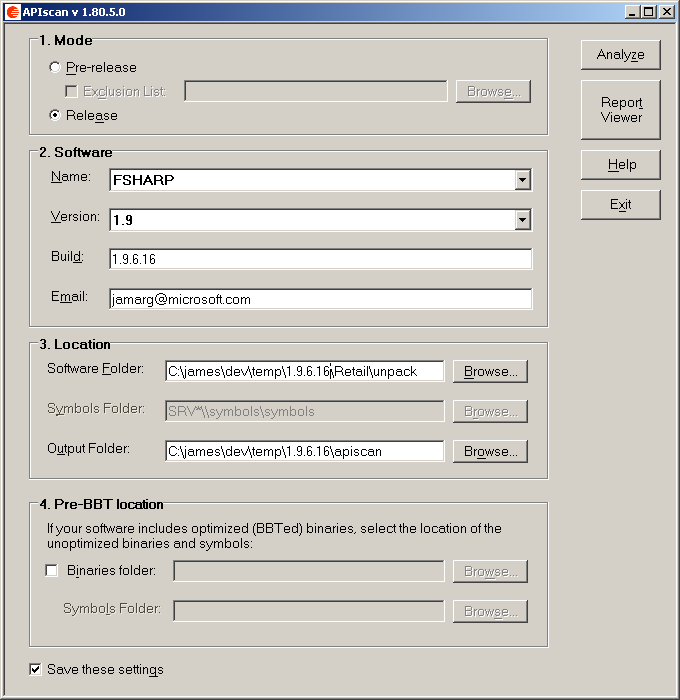
We should have a LCA valid license for this release. I asked “John Mulgrew (LCA)” if we need to check for each new release. As I understand it, provided the release is an incremental update with no significant new functionality then rechecking with LCA is not required.

## Policies: Interoperability

### Windows API Usage

APISCAN should be run on all the released files. I run it on the “unpack” directory (copied local from the build machine) which also contains copies of the pdbs which are needed. It writes to an output directory. One dialog needs completing, then Analyze, then View Report. Note “HELP” has error code reference.

NOTE: Select name=FSHARP and not F# (the hashes break the result URLs).



### Launched or embedded Windows

Not applicable to us, so OK.

### Protocol Documentation

Not applicable to us, so OK.

### Protocol Inventory

Not applicable to us, so OK.

## Policies: Privacy

### Privacy Statement

Currently we do not transfer any user data (for example their IP address) to MS or 3rd-parties. As such we do not require privacy statement or data transfer consents, so OK.

### Consent for data transfers

OK.

## Policies: Security

### Export Controls

F# does not use cryptography and is not configured for military use etc, so we are OK here.

### SDL

The project is registered at <http://switrack/>.

We need a Switrack project id and permissions to complete CPX. Prior to 1.9.6.0 we ticked “MSR Download” and had exemption from completing this section. For 1.9.6.0 we completed the (SDL?) process. 1.9.6.2 was a bug fix update for 1.9.6.0.

If we need to repeat this process, it takes time…

## Policy: Software integrity

### Virus Scanning

“make cpx” creates an unpack directory containing zip, msi and their contents, e.g.

FSharp-1.9.6.16-CPx\Retail\unpack

This can be shared and submitted for virus check at Dublin or Redmond using:

<http://vcs/>

Confirm the job scanned clean (an email arrives) and the tool used as <http://vcs>.

### Code-signing

Our .NET assemblies are strong signed. Additionally, all our DLL, EXE and MSI need to have Microsoft Digital Signatures via <http://codesign> – which is automated with a “signed build”.

The files to check are:

* InstallFSharp.msi and
* DLL and EXE contained inside the MSI.

The “unpack” directory contains the MSI and the contents of the MSI under “unpack/msidump”. Checking “file properties” and “digital signature” TAB will confirm that code-signing did occur.

## Doing a signed build

Requirements:

1. You need CODESIGN Submitter Object installed on the build machine.

[\\csneovlt.redmond.corp.microsoft.com\public\Submitter Tool for Download](file:///\\csneovlt.redmond.corp.microsoft.com\public\Submitter%20Tool%20for%20Download)

1. You need your user account activated for code-signing.

<http://codesigninfo/Wiki/Getting%20Started.aspx>

1. You need a Smart Card and reader. These permissions can pass over remote desktop.

Pre-build checks:

1. Visit <https://codesign.gtm.microsoft.com/>

From DIAGNOSTICS tab do “VERIFY INSTALLATION” and “VERIFY SMART CARD”.

1. make submitter-test

This does a “test submission” on a single exe.

You will need to authorize it with your Smart Card and PIN. No need to approve it.

It is better to find any setup/smartcard/submissions failures upfront!

Make a signed build.

1. users\matteot\sync\_build\_publish\_doit.cmd “CODESIGN=yes”
2. One hour into the build the codesign submissions start.

Submission requires a smartcard and the user to enter their PIN in the dialog box, so …

<BEGIN: manual interaction>

1. Approve the first round submissions (about 7-8 directories).
2. Then get these approved by 2 of the approvers (aliases listed in Makefile).
3. Then wait another 10 minutes for the MSI to build.
4. Then approve the second round submission (just the msi)

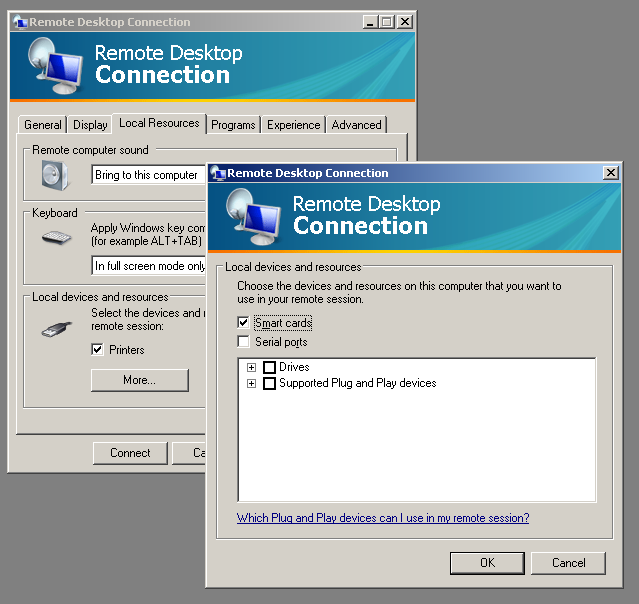
<END: manual interaction>

1. When this is approved the build will run through to completion.
2. Check build completed OK.
3. Check the final robocopy to remote shares was OK.

If not, fix up permissions and copy the drops in .\LocalDrop to the remote locations.

1. Run “make cpx”.

## Smart Card sent over Remote desktop.

If smart card is not recognized on the remote machine, check the RDP options. 

To verify, lock the remote machine, is there is a smart card option under “other credentials” to unlock it?