Ecole d'ingénieurs et d'architectes de Fribourg
Hochschule für Technik und Architektur Freiburg

ViSaG - Virtual Safe GRID

Virtual - POP-C++

Road to ViSaG

Author:
Valentin Clément

Date: December 15, 2010
Revision: 1.1

Contents

1 Introduction 2

2 Requirements for Virtual POP-C++ 2
2.1 Hypervisor related data 2
2.2 Code injection in wrapper 2
2.3 POP Application Identifier 2
2.4 Installation of admin and worker VM 3
2.5 Cloning VM 3
2.6 Preparation of the worker VM 4
2.7 JobMgr - Management of VM and jobs 4
2.8 WorkerDeamon 4
2.9 Pseudo-Main 5
2.10 Fusion of POP-C++ 1.3.1 beta JS1.2 and Virtual-POP-C++ 5
2.11 Minimal Linux distribution 5
2.12 Add restrictions on SSH 5
2.13 ESXi user and restriction 6
2.14 Testing 6
2.15 Encrypted file system 6

3 Repartition 7

4 Open points and information 8
4.1 VMware VIX API 8
4.2 Cloning 8
4.3 Key exchange with the VM 9
4.4 Names and addresses used in Virtual POP-C++ 9
4.5 DHCP 10

5 Modifications - *DRAFT* 11
5.1 Ability to compile two different versions of POP-C++ 11
5.2 Hypervisor information 11
5.3 POP Application Identifier 11
5.4 POP-C++ and VIX compilation 12
5.5 Using VIX to get the worker VM IP address 12
5.6 Using VIX to send and write the admin VM public key on worker VM 13
5.7 Cloning function implementation 13
5.8 Requirements 13

6 Ability to manage more than one VM 13

7 Glossary 14

8 Table of figures 15

9 References 15

1 Introduction

Virtual - POP-C++ is a version of the POP-C++ middle-ware that would be able to run parallel objects into virtual machines. The development of this version is the core of the ViSaG project. A prototype version of Virtual - POP-C++ has been developed during a bachelor thesis project in the summer 2010. This version is called VirtualPOPC-1[1].

NOTE : VirtualPOPC-1 will be referred as VPOP1 in the rest of this document.

2 Requirements for Virtual POP-C++

This chapter aims to identify the weaknesses and the missing elements of VPOP1. At the end of this chapter, all the modifications that have to be brought for the final version to be produced for the ViSaG project will be summarized and shared between the EIA-FR and the HEPIA.

2.1 Hypervisor related data

In VPOP1, all the data related to the hypervisor connection are saved as environment variables. These data should be saved in the JobMgr config file located at *POPC_LOCATION*/etc/jobmgr.conf. The hypervisor password should be encrypted and not saved in clear.

Files to be modified : ./lib/jobmgr.cc, ./include/jobmgr.ph, ./script/popc_setup.in

Work done:
All the hypervisor related data are now stored in a file located in the POP_LOCATION/etc/virtual.conf. These data could evolve with the changes that will be done in the future.

Decision:
The password is not encrypted yet. This point is still open at the moment (a discussion with security expert must be held to find the best practice to put in place).

2.2 Code injection in wrapper

In the wrapper used to contact the hypervisor, some methods execute a shell command. As the command executed is prepared in the method, some malicious codes could be injected. To avoid this code injection, the commands should be controlled before their execution.

Files to be modified : ./lib/ESXWrapper.cc

Work done:
Nothing for the moment.

Decision:
The methods using shell command will be replaced by VIX API code. The cloning process could run some external shell command but it is not well defined for the moment.

2.3 POP Application Identifier

A POP Application Identifier must be created for any POP-C++ application. This ID must be sent with the resource discovery request. A virtual node will create only one virtual machine (VM) per application. Any parallel object of the same application running on the same virtual node will be executed in the same VM. This VM will be shutdown at the end of the application or after a specified idle time.

File to be modified : ./lib/request.cc, ./include/request.h, ./lib/jobmgr.cc, ./lib/appservice.ph, ./lib/appservice.cc, ./lib/popc_search_node.cc

Work done:
The POPAppId is passed in the resource discovery process and the admin-VM takes it into consideration for the workers management. The VM is shutdown when it has no more jobs running on it. This step is finished.

Decision:
The VM could be shutdown when the application is finished. A message must be send to all nodes to shutdown the VM associated with the application.

2.4 Installation of admin and worker VM

The installation of the environment on the admin and worker VM could be the source of many problems. This installation should be as automatic as possible (missing library, ssh key generation, ssh_config, cloning VM ...).

Files to be modified : ./script/popc_setup.in, ./configure.ac

Work done:
The configure script is no able to accept the option "–enable-virtual". This option allows the user to compile POP-C++ in the Virtual version. Without this option, the standard version is compiled. To be able to compile two different versions, a VirtualJobMgr and a VirtualPOPCSearchNode have been implemented. All the virtual version is implemented by overwriting basic function of the standard JobMgr and POPCSearchNode. This step is almost done. There is some work to do in the installation script (remove useless questions, check SSH, generate key if not present ...).

Decision:
Do the small modifications on the popc_setup script as soon as we have the time.

2.5 Cloning VM

The worker VM must be cloned as much as needed on a virtual node. This process should be done before needing the VM. As the cloning process takes some time, it could be a good idea to start the cloning process when just one VM is available.
For example, a node running Virtual POP-C++ is using 3 workers. When the second worker is reserved (just one more worker free), the cloning process starts to make a new worker. The maximum number of worker on a node must be defined during the installation.

Files to be modified: Not defined

Work done:
Some experiments with VIX and with a home made script. Nothing implemented in POP-C++ for the moment. HEPIA has maybe some hints to give about this process.

Decision:
Using the hard copy solution from a C++ code. Asking HEPIA if they could find a better solution. The cloning process starts when there are only X free VM left. The number of VM prepared in advance is specified as an option.

2.6 Preparation of the worker VM

In the current version, the preparation of the VM is not very reliable. In fact, during this preparation, the ESXWrapper will try to get the VM IP. After three attempts, the preparation will fail if the IP is not discovered. This process should be optimized and maybe done differently.

Files to be modified : ./lib/ESXWrapper.cc

Work done:
Solved the waiting problem. Try with the VIX API, which seems to be more reliable than the command line tool. Nothing is implemented with the VIX API in POP-C++ for the moment but I got a test program sample.

Decision
Using the VIX API to retrieve the IP address. DHCP renewal must be done before retrieving the address (this point is open because more information on DHCP must be found).

2.7 JobMgr - Management of VM and jobs

As the JobMgr running on the admin-VM must be able to manage several worker-VM, this JobMgr must be modified to include the management of different VM with the POPAppID. This management must include the startup, check during runtime and shutdown of a VM for an application.

Files to be modified : ./lib/jobmgr.cc, ./lib/ESXWrapper.cc

Work done:
Modify the check and reserve process for taking into consideration the management of several workers. Ability to run several objects of the same application on the same VM. Host resource management must be defined.

Decision:
The admin-VM manage all the resource on the node. The all node is seen as one POP-C++ node. The VMs are just a way to encapsulate the parallel objects execution.

2.8 WorkerDeamon

The WorkerDeamon is in charge of the key exchange between the admin-VM and the worker-VM. This worker is not well developed. Here are some hints to find a different way to exchange the keys:

Hints:

- The keys could be passed during the installation of the environment. Once one worker-VM is cloned, its virtual disk could be mounted on the admin-VM and the admin-VM key could be write in the file system.
- The keys could be passed during the installation of the environment. Once one worker-VM is cloned, it could be started and the keys exchanged before taking a snapshot of it.
- A POP-C++ parallel object could be used to do that. The creation of a virtual POP-C++ service as a parallel object could run independently on the worker-VM. This parallel object could be contacted by the admin-VM and the keys could be exchanged.

Work done:
Test program with the VIX API and the copy file function. This solution seems to work perfectly. With

this solution, the POP-C++ Security Manager running on the admin-VM could manage the keys of all workers. Nothing has to run on the worker.

Decision:
Use the VIX API to do this task. Ask HEPIA is they could find another way to communicate between the admin-VM and the worker-VM. We are aware that this solution is less generic.
Some other solution must be explored such as shared disk or shared network (libvirt).

2.9 Pseudo-Main

As we want to run a POP-C++ application in a full secured environment, the main of the application should be launched in a VM too. To do that, a pseudo-main could redirect the real main on a VM and launch it in a secure way.

Files to be modified : POP-C++ compiler (paroc_main)

Work done:
Nothing for the moment.

Decision:
This point is open. We can offer an option to let the user choose to run the main in a VM or not.

2.10 Fusion of POP-C++ 1.3.1 beta JS1.2 and Virtual-POP-C++

The Virtual version of POP-C++ must be merged with the secure version of the POP-C++. This will make the ViSaG version of POP-C++.

Work done:
Prepare the configure script to be able to accept the option "–enable-secure".

2.11 Minimal Linux distribution

The admin-VM should be as light as possible. In fact, this admin-VM will never run a parallel object for a POP-C++ application. This VM is only in charge of the management of the others VM known as worker-VM. Due to this fact, this VM should run the lightest distribution of Linux that can run the POP-C++ Global Services. This distribution should also support "libvirt" and the "vSphere CLI" libraries.

Work done:
Nothing for the moment. HEPIA could work in this.

2.12 Add restrictions on SSH

It's possible to add restrictions on the use of SSH for a specific public key. It could be a good idea to allow only what we really need for POP-C++ (SSH tunnelling and the popcrun command). This restriction are added with the public key in the "authorized_keys" file. The POP-C++ Security Manager should be able to add these restrictions when it writes a new key.

We can imagine that all SSH restrictions are defined during the installation of POP-C++. These information will be passed to the PSM for future usage.

Work done:
Nothing for the moment.

Decision:
Add SSH restrictions. Only enable the tunnelling and the use of the "popcobjrun" command.

2.13 ESXi user and restriction

VMware ESXi has a user and group management system. It would be more secure to create a specific user for Virtual POP-C++ and allow only the specific action related to its usage.

The administrator of the ESXi platform has many privileges that can be granted or revoked. All the actions about users and groups must be done from the vSphere Client.

Work done:
Nothing for the moment.

Decision
Using the root account for the moment. When the full version of POP-C++ Virtual and Secure will be ready, ask HEPIA to do some test if we can reduce the rights and use a different account.

2.14 Testing

A full protocol of test must be done on the full version of POP-C++ for ViSaG. The tests must contain :

1. Stress test to check the concurrency problems.
2. Performance test to be able to measure the overhead and make some optimizations.
3. Memory test using mudflap or valgrind to check any memory leaks.

2.15 Encrypted file system

The possibility to use an encrypted file system on the VM could enforce the security of the whole project. This point is open and must be studied.

3 Repartition

This section suggests a repartition of the work between the EIA-FR and the HEPIA (the two schools working on the ViSaG project and more specifically on the lower level of this project).

Theoretical, the EIA-FR is in charge of the major modifications of the POP-C++ middle-ware and the HEPIA is in charge of the global VM management. The task above will be

Section	**What ?**	**Who ?**	**Priority**
2.1	Hypervisor informations	EIA-FR	Medium (Done, except password crypt)
2.2	Code injections in wrapper	EIA-FR	Medium
2.3	POP Application Identifier	EIA-FR	(DONE!)
2.4	Installation of admin and worker VM	EIA-FR	(DONE!)
2.5	Cloning VM	HEPIA (EIA-FR)	On the way
2.6	Preparation of worker VM	EIA-FR	(Will be done with VIX)
2.7	JobMgr - management	EIA-FR	(on the way)
2.8	WorkerDeamon	EIA-FR	High (VIX API)
2.9	PseudoMain	EIA-FR	Medium
2.10	Fusion	EIA-FR	High
2.11	Minimal Linux distribution	HEPIA	Medium
2.12	Add restrictions to SSH	EIA-FR	High
2.13	ESXi user and restrictions	HEPIA	Medium
2.14	Testing	HEPIA and EIA-FR	High (wait full version)
2.15	Encrypted file system	HEPIA and EIA-FR	Medium

4 Open points and information

This chapter regroups some points that are still open and some useful information for taking decisions about the future step to do in the ViSaG project. Some of those points have been discussed and a decision has been made (see Chapter 2 for the decisions made).

4.1 VMware VIX API

VMware provides a C based API to interact with VM. This API has some interesting function that could help us in the development of the Virtual version of POP-C++. These functions are:

- **VixVM_Clone** : cloning an existent VM with snapshot (unimplemented with ESXi).
- **VixVM_CopyFileFromGuestToHost** : copy a file from the guest VM (worker) to the host VM (admin)
- **VixVM_CopyFileFromHostToGuest** : copy a file from the host VM (admin) to the guest VM (worker)
- **VixVM_ReadVariable** : some property of the VM (ip address) could be read like that.

The VIX 1.10 API reference can be found at following URL:
http://www.vmware.com/support/developer/vix-api/vix110_reference/

NOTE: The "product support" section of the reference cited above mentions that the function VixVM_Clone is only supported on VMware Workstation platform.

REMARK: Using a proprietary API could be a little bit restrictive but it would also be much more reliable than the use of command line tools that are also proprietary.

4.2 Cloning

The cloning process of a VM is going like this :

1. Create a new folder on the data store (vifs –mkdir).
2. Copy every files from the original VM folder to the new folder (except log files) (vmkfstools -i).
3. Modify the "display name" of the VM (vifs -g, awk, vifs -p).
4. Register the new VM in the inventory (vmware-cmd).
5. Start the VM (vmware-cmd).

This solution could be easily implemented in the current version of Virtual POP-C++ (a shell script is already working for that purpose).

NOTE(1): None of the VIX API or LIBVIRT is able to provide this function for the moment.

NOTE(2): When a worker is cloned, the SSH pairs is the same on the original VM and on the cloned VM. This is not a problem for the functionality of POP-C++. It could be a bad practice from a security point of view. We could force the worker to generate new key after the cloning process by launching a command with VIX.

4.3 Key exchange with the VM

The workerDeamon must be changed because now it is not reliable and not secure enough. In addition, with the secure version, the worker VM must be able to do more than just receive a key. Here are the needs for the key exchange:

- The admin must be able to read the PKI of the worker.
- The worker must accept PKI from the admin.

The VIX API gives a way to copy a file from a host to a guest. This API also gives a way to execute a program on the guest.

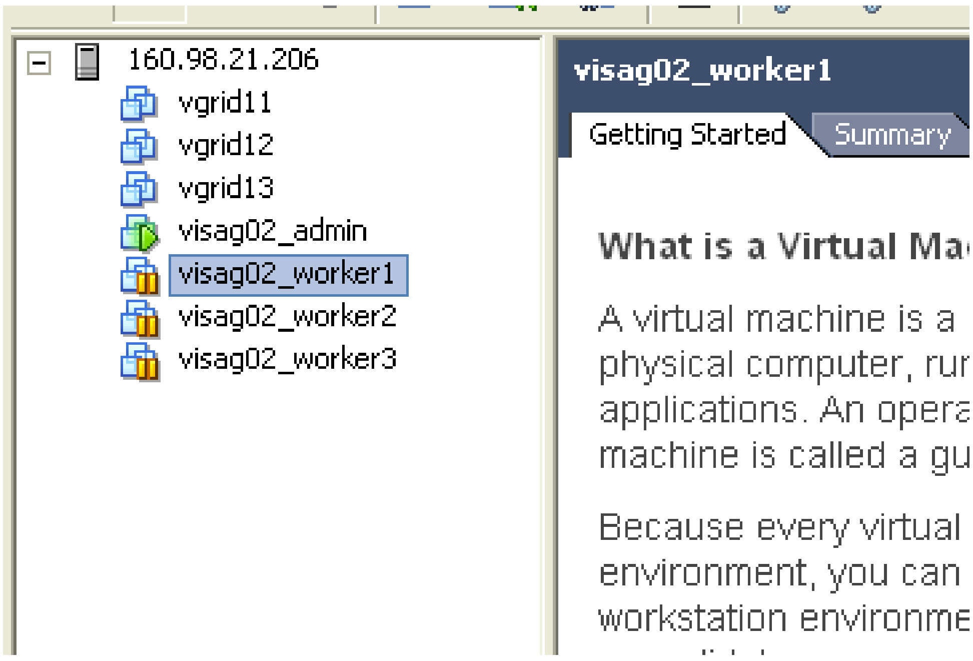
4.4 Names and addresses used in Virtual POP-C++

POP-C++ and its virtual version use different mechanism to retrieve a machine identity. Here is a brief summary of methods used in the virtual version of POP-C++ currently in development.

Between Global Services
Between the global services, either the IP address or the hostname could be used.

Between VM-admin and VM-worker
Between the VM-admin and the VM-worker two different kind of identification are used. Firstly, when the admin reverts the worker to the correct snapshot, the "display name" of the VM is used. This "display name" is a notion from VMware ESXi. A VM is registered under this name in the vSphere inventory (see Figure 1). With this "display name", the admin is able to retrieve the IP address of the worker. Once the admin knows the worker IP address it can communicate directly with the worker (at this point the hostname could also be used).

Figure 1: vSphere Client - inventory (display name)



What is a Virtual Ma

A virtual machine is a
physical computer, rur
applications. An opera
machine is called a gu

Because every virtual
environment, you can
workstation environme

4.5 DHCP

Using the DHCP service for the worker is not without any risks. This section will list the known issues about its usage.

Lease on a suspended VM
DHCP service uses lease for IP address. When a VM is reverted to a snapshot from a "shutdown state" or a "suspended state", the lease could be expired and the IP address may change after a few minutes. This behaviour could make the POP-C++ application crash. A solution to this problem is to force the VM to ask for a new lease when it is reverted. In this case the IP address would be changed before any execution on the worker and the application could run normally.

5 Modifications - *DRAFT*

This chapter is a review of the modifications made on the prototype to make a fully functional version of Virtual POP-C++ for the ViSaG project.

5.1 Ability to compile two different versions of POP-C++

As the Virtual version of POP-C++ is just installed on the admin-VM, the ability to compile a standard version (without virtual modifications) is a great choice for the end user.

To make this optional compilation possible, two objects have been created:

- VirtualJobMgr: this object inherits from the JobMgr and overwrites the needed methods.
- VirtualPOPCSearchNode: this object inherits from the POPCSearchNode and overwrites the needed methods.

All the modifications related to the virtual version are now made in the VirtualJobMgr and in the VirtualPOPCSearchNode. A VirtualPOPCSecurityManager will certainly be needed. This modification will let the future version of POP-C++ evolve on the same basis and the merging needs will be less important.

Files created: ./lib/virtual_jobmgr.ph, ./lib/virtual_jobmgr.cc, ./lib/virtual_popc_search_node.ph, ./lib/virtual_popc_search_node.cc

Files modified: ./configure.ac, ./lib/jobmgr.ph, ./lib/jobmgr.cc, ./lib/popc_search_node.ph, ./script/popc_setup.in

FULL EXPLANATION ...

5.2 Hypervisor information

Instead of being stored in environment variable, the information relative to the hypervisor connection have been stored in the VirtualJobMgr configuration file. This file is located under *POPC_LOCATION*/etc/virtual.conf. The installation script is now writing the information directly into this configuration instead of writing them into a script that initialize the environment variables.

The JobMgr constructor retrieves all the information stored in the configuration file. These values are stored into attributes and used when needed.

Files modified : ./lib/virtual_jobmgr.cc, ./lib/virtual_jobmgr.ph, ./script/popc_setup.in

5.3 POP Application Identifier

The AppCoreService has been modified to generate the POP Application ID. This ID is generated in the constructor of this object. The "Request" object has also been modified to include this ID. Just before launching the resource discovery, the JobMgr will contact the AppCoreService to know the POPAppID and set it in the request. The other JobMgrs contacted will know for which application a request has been launched.

For more information about the POP Application Identifier, please refer to the document "POP-C++ over SSH Tunnel"[2] in section 7.1.

In addition to these modifications, the reservation process must also include the POPAppID. The "struct Resources" have been modified to include the POPAppID. This struct is defined in the JobMgr header file.

The "Reserve" method must now accept the POPAppId as a parameter. At this point, the JobMgr has all the information to manage the different VM worker.

Files modified : ./include/request.h, ./include/Makefile.am, ./lib/request.cc, ./lib/appservice.ph, ./lib/appservice.cc, ./lib/jobmgr.cc, ./lib/Makefile.am

Files added : ./include/popwayback.h, ./lib/popwayback.cc

5.4 POP-C++ and VIX compilation

Some modifications have to be made to be able to compile POP-C++ with the VIX library. The modifications are listed below :

File: configure.ac:
As the VIX API does not install a ".pc" file for the pkg_config function. We have to declare the _CFLAGS and the _LIBS marco for VIX. The lines below have been added to the "configure.ac" file.

```
VIX_CFLAGS="-lvixAllProducts -ldl"
VIX_LIBS="-I/usr/include/vmware-vix"
AM_SUBST([VIX_CFLAGS])
AM_SUBST([VIX_LIBS])
```

File: ./lib/Makefile.am:
The input Makefile located in the "lib" directory must also be modified. The VIX macros must be added to the macro "AM_CXXFLAGS".

```
AM_CXXFLAGS= OTHER_FLAGS $(VIX_CFLAGS) $(VIX_LIBS)
```

With those modifications, POP-C++ can now be compiled with the VMware VIX API.

NOTE: The VIX API must be installed on the machine which compile POP-C++ in its Virtual version.

5.5 Using VIX to get the worker VM IP address

The VIX API allow us to read some properties on the guest VM. This ability is used to read the IP address of the guest VM. The process is going as follows:

1. Connect to the hypervisor (VixHost_Connect())
2. Get a pointer to the VM (VixHost_OpenVM())
3. Power on the VM (VixVM_PowerOn())
4. Waiting for the VMware tools to be ready (VixVM_WaitForToolsInGuest())
5. Read the variable "ip" (VixVM_ReadVariable())

NOTE: For the full code, please see the file ./lib/ESXWrapper.cc method "_popc_domainGetIpAddress(POPvm &vm)"

5.6 Using VIX to send and write the admin VM public key on worker VM

The VIX API has two very interesting function that allow us to send a file on the worker VM and to execute a command on the worker VM. We have to send the admin VM public key and write it in the authorized_keys file on the worker VM. The process to do this using VIX is going as follows:

1. Connect to the hypervisor (VixHost_Connect())
2. Get a pointer to the VM (VixHost_OpenVM())
3. Power on the VM (VixVM_PowerOn())
4. Waiting for the VMware tools to be ready (VixVM_WaitForToolsInGuest())
5. Login in the guest VM (VixVM_LoginInGuest())
6. Copy the admin public key file on the guest VM (VixVM_CopyFileFromHostToGuest())
7. Execute a command on the guest to write the key (VixVM_RunProgramInGuest())

NOTE: For the full code, please see the file ./lib/ESXWrapper.cc method "_popc_sendLocalPublicKey(POPvm &vm)"

5.7 Cloning function implementation

As no built-in cloning function has been found either in the VIX API or in libvirt, this function has been implemented from scratch. The wrapper method "_popc_domainClone(POPvm &original, POPvm &clone)" does the whole process. As this process could take sometimes, this function will create a child process.

5.8 Requirements

To be able to run the POP-C++ Virtual version, the following requirements must be fulfilled:

VM Type	POP-C++ version	VMware tools	VMware CLI	VIX	libvirt
Admin	Virtual Secure	X	>= 4.1	>= 10.1	>= 0.8.5
Worker	Secure	X	-	-	-

- VMware ESXi 4.1 or later must be installed as the hypervisor.
- The VMware tools 8.3.2 build-257589 or later must be installed on the worker VM and on the Admin VM.
- The VIX API 1.10 or later must be installed on the admin VM.
- libvirt 0.8.3 or later must be installed on the admin VM.

6 Ability to manage more than one VM

In the first prototype of VPOP-C++, the JobMgr was able to manage only one VM. In the final version, the JobMgr must be able to manage many VM. To enable this ability, some change have been made in the Wrapper interface. The wrapper interface was designed for one VM only. A new object called "POPvm" is now holding all the information about a specific VM. This object is passed through the Wrapper. Additional information can be inserted in the "POPvm" object without modifying the "Wrapper" (could be done for different virtualization platform than ESXi).

7 Glossary

- **API** : Application Programming Interface
- **VIX** : API of VMware to interact with the VMware hypervisor (or Workstation ...)
- **EIA-FR** : College of Engineering and Architecture of Fribourg
- **HEPIA** : Haute Ecole du Paysage, d'Ingénierie et d'Architecte de Genève
- **POP-C++** : Parallel Object Programming C++
- **ViSaG** : Virtual Safe GRID
- **VM** : Virtual Machine

8 Table of figures

1 vSphere Client - inventory (display name) 9

9 References

[1] Adrian Wyssen, *VirtualPOPC-1 : Project Report*. EIA-FR, Switzerland, June-August 2010.

[2] Valentin Clément, *POP-C++ over SSH Tunnel*. EIA-FR, Switzerland, September-November 2010.