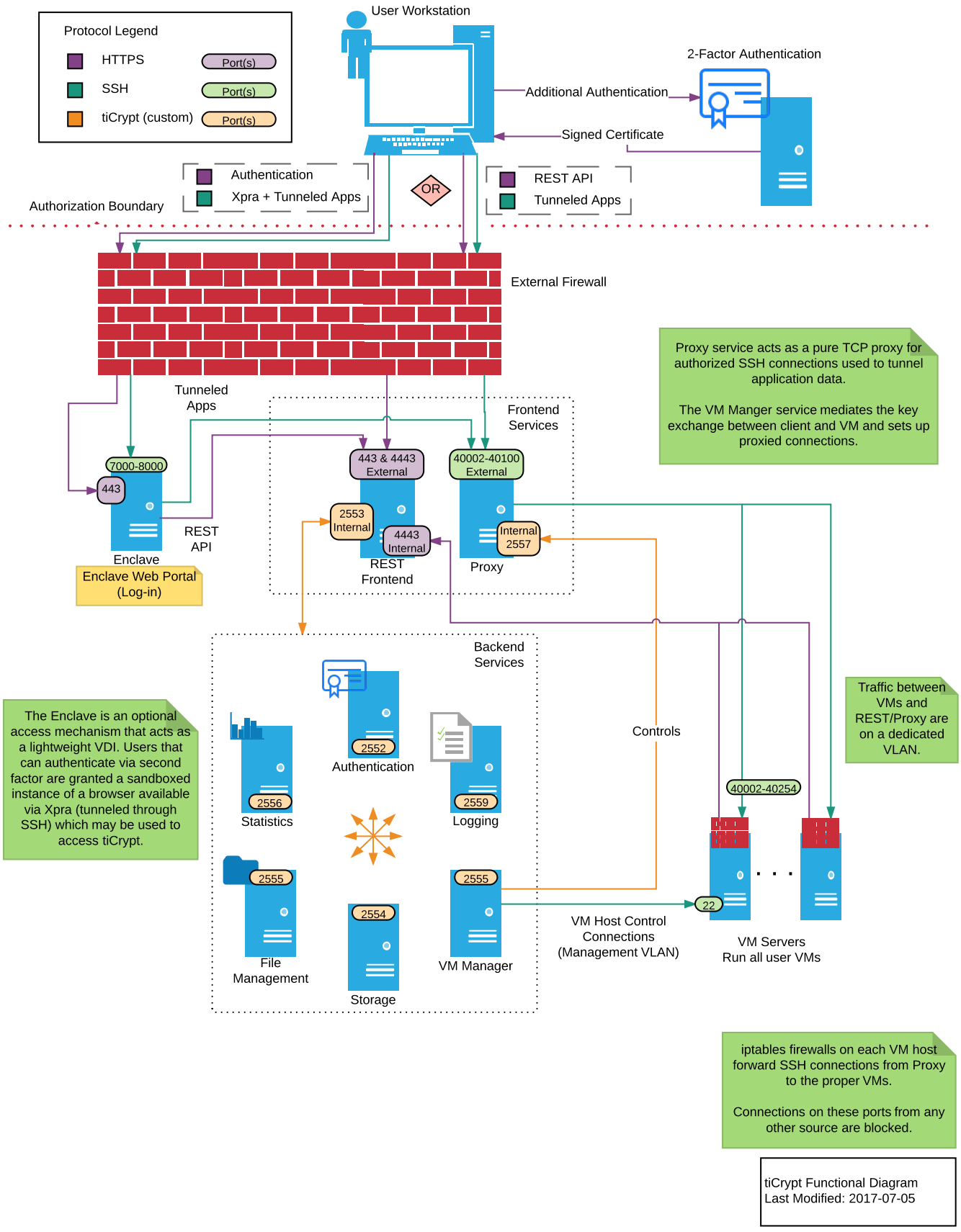
tiCrypt ARCHITECTURE PART ONE

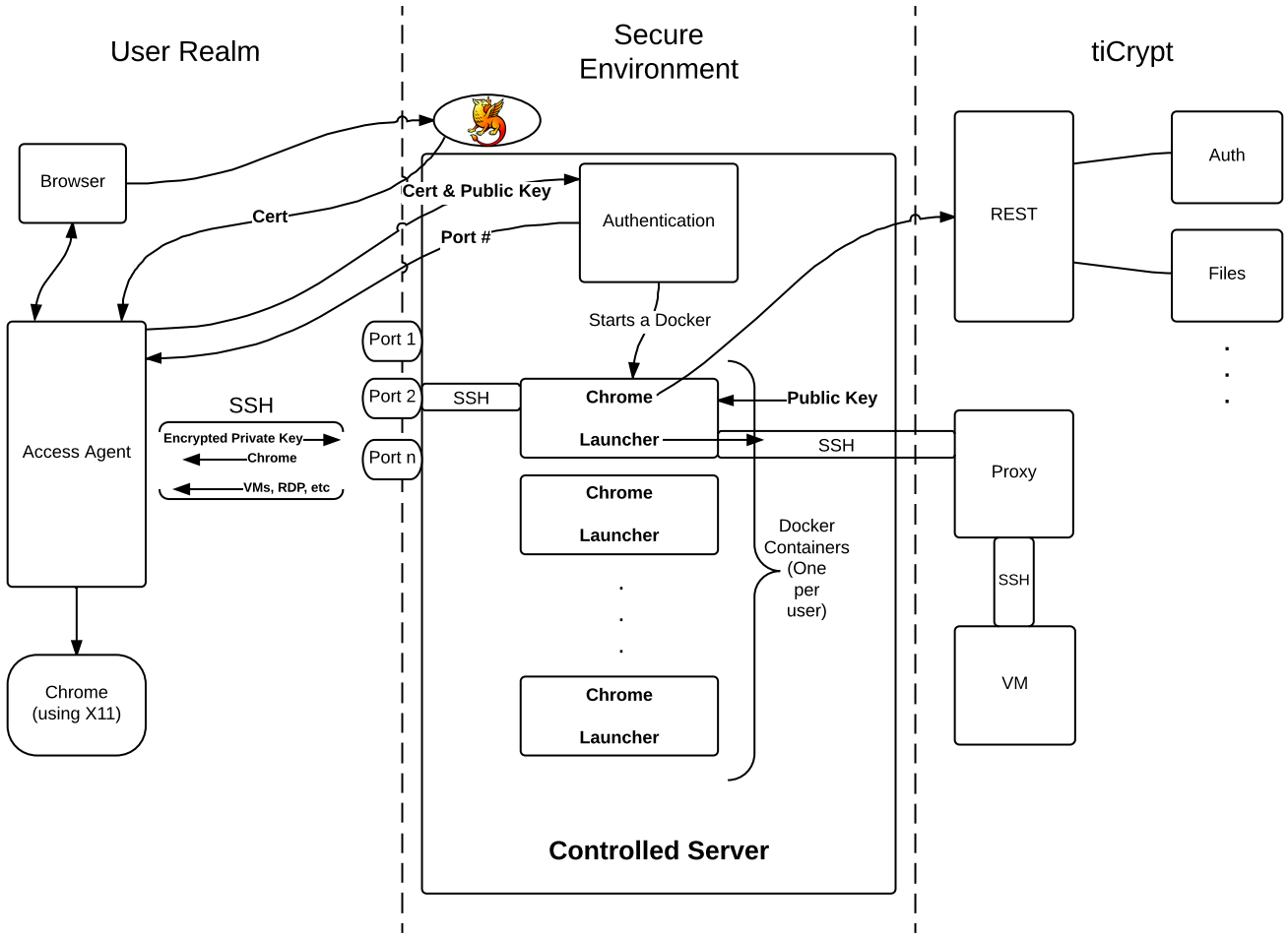
Security Environment, Functional Diagram, and VM Lifecycle

THOMAS SAMANT
thomas@terainsights.com

ALIN DOBRA
alin@terainsights.com

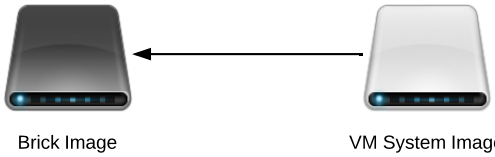
Tera Insights, LLC
Feb 22, 2018


Protocol Legend
HTTPS
Port(s)
SSH
Port(s)
tiCrypt (custom)
Port(s)
User Workstation
2-Factor Authentication
Additional Authentication
Signed Certificate
Authentication
Xpra + Tunneled Apps
OR
REST API
Tunneled Apps
Authorization Boundary
External Firewall
Tunneled Apps
Frontend Services
Proxy service acts as a pure TCP proxy for authorized SSH connections used to tunnel application data.
The VM Manger service mediates the key exchange between client and VM and sets up proxied connections.
7000-8000
443
Enclave
REST API
Enclave Web Portal (Log-in)
443 & 4443 External
2553 Internal
4443 Internal
REST Frontend
40002-40100 External
Internal 2557
Proxy
Backend Services
The Enclave is an optional access mechanism that acts as a lightweight VDI. Users that can authenticate via second factor are granted a sandboxed instance of a browser available via Xpra (tunneled through SSH) which may be used to access tiCrypt.
2552
Authentication
2556
Statistics
2559
Logging
2555
File Management
2554
Storage
2555
VM Manager
Controls
Traffic between VMs and REST/Proxy are on a dedicated VLAN.
40002-40254
22
VM Host Control Connections (Management VLAN)
VM Servers
Run all user VMs
iptables firewalls on each VM host forward SSH connections from Proxy to the proper VMs.
Connections on these ports from any other source are blocked.
tiCrypt Functional Diagram
Last Modified: 2017-07-05


User Realm
Secure Environment
tiCrypt
Browser
Cert
Cert & Public Key
Port #
Authentication
REST
Auth
Files
Starts a Docker
Port 1
Port 2
Port n
SSH
Chrome
Launcher
Public Key
SSH
Access Agent
SSH
Encrypted Private Key
Chrome
VMs, RDP, etc
Chrome
Launcher
Chrome
Launcher
Docker Containers (One per user)
Proxy
SSH
VM
Chrome (using X11)
Controlled Server

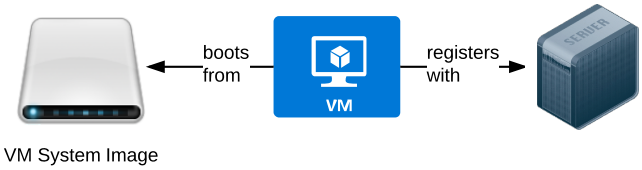
Virtual Machine Lifecycle

Stage 1: Operating System Image Creation



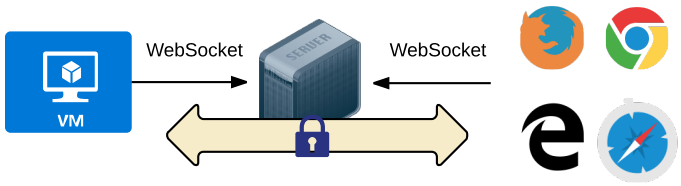
The VM's Operating system image is a thin, "copy-on-write" volume that keeps track of any differences between the VM's current state and the original Brick image. This ensures that every VM starts with the exact same initial state, and that VMs do not interfere with each other.

Stage 2: VM Creation



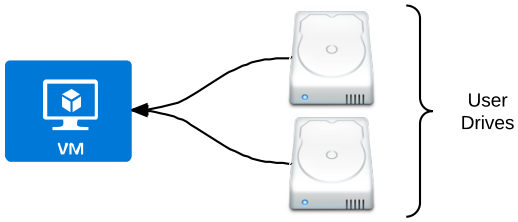
A new Libvirt domain is started with the image created in Stage 1 as the main disk. When basic initialization is complete, the VM sends a message to the server to notify it that it is ready. This message also contains some information about the VM, such as its IP address and its public key. In return, the server gives the VM the public key of the user that created it.

Stage 3: WebSocket Connection Establishment



The VM and browser both open WebSocket connections to the server, which joins the connections together. Using the public keys that were exchanged in the previous step, the VM and the browser establish an encrypted channel over the WebSocket connection. This encrypted channel is used by the browser to issue commands to the VM.

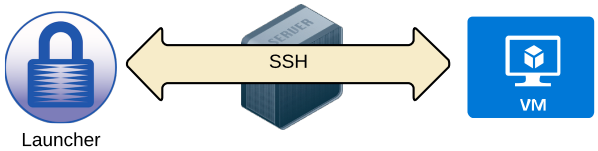
Stage 4: Drive Attachment



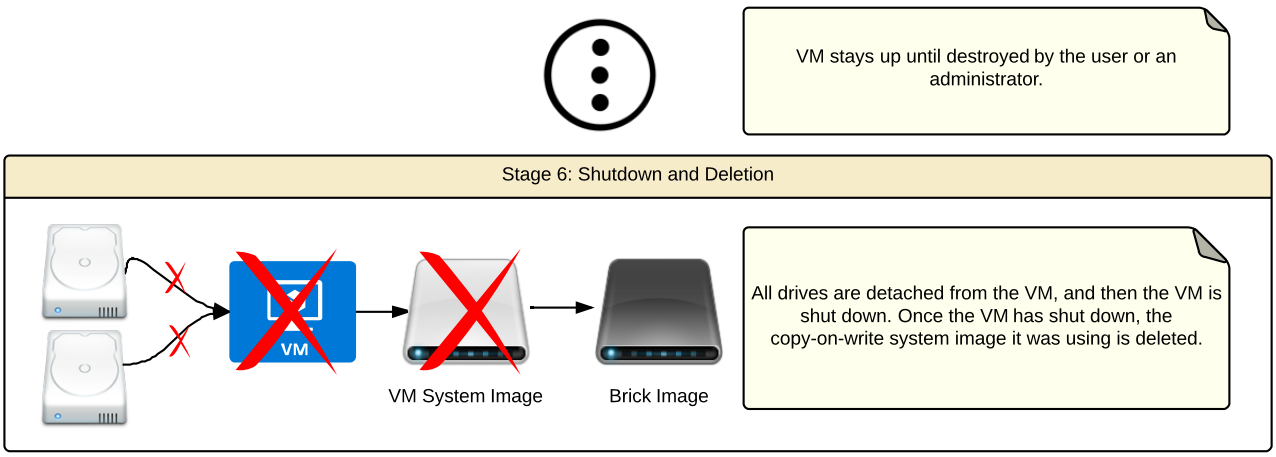
With the WebSocket connection established, any drives the user had selected for use with the VM are attached. The necessary encryption keys needed to unlock the encrypted drives are transmitted over the WebSocket connection.

At this point, the VM is considered "ready", and non-graphical VMs are ready for use immediately.

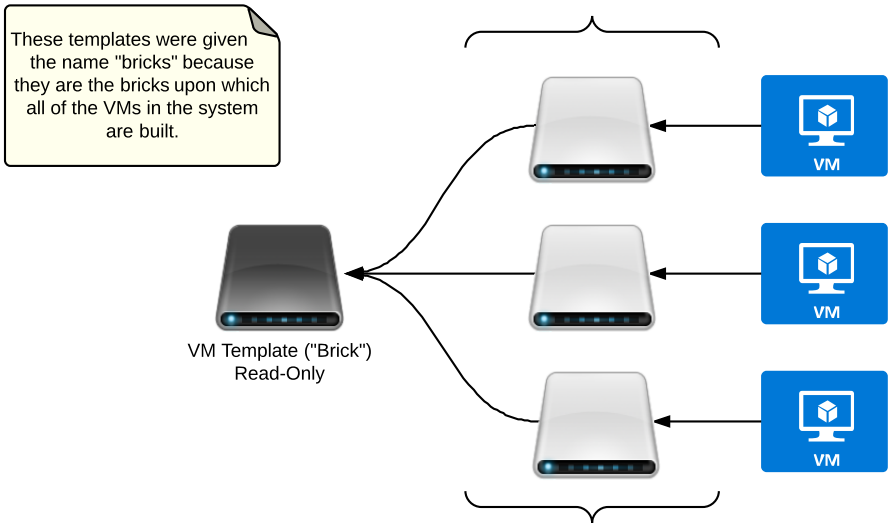
Stage 5: SSH Proxy (Graphical VMs only)



The server acts as a proxy for the SSH bridge between the launcher and the VM. This allows the VM to securely provide arbitrary services to the client. When establishing the SSH connection, the client and VM exchange keys through the encrypted WebSocket channel.

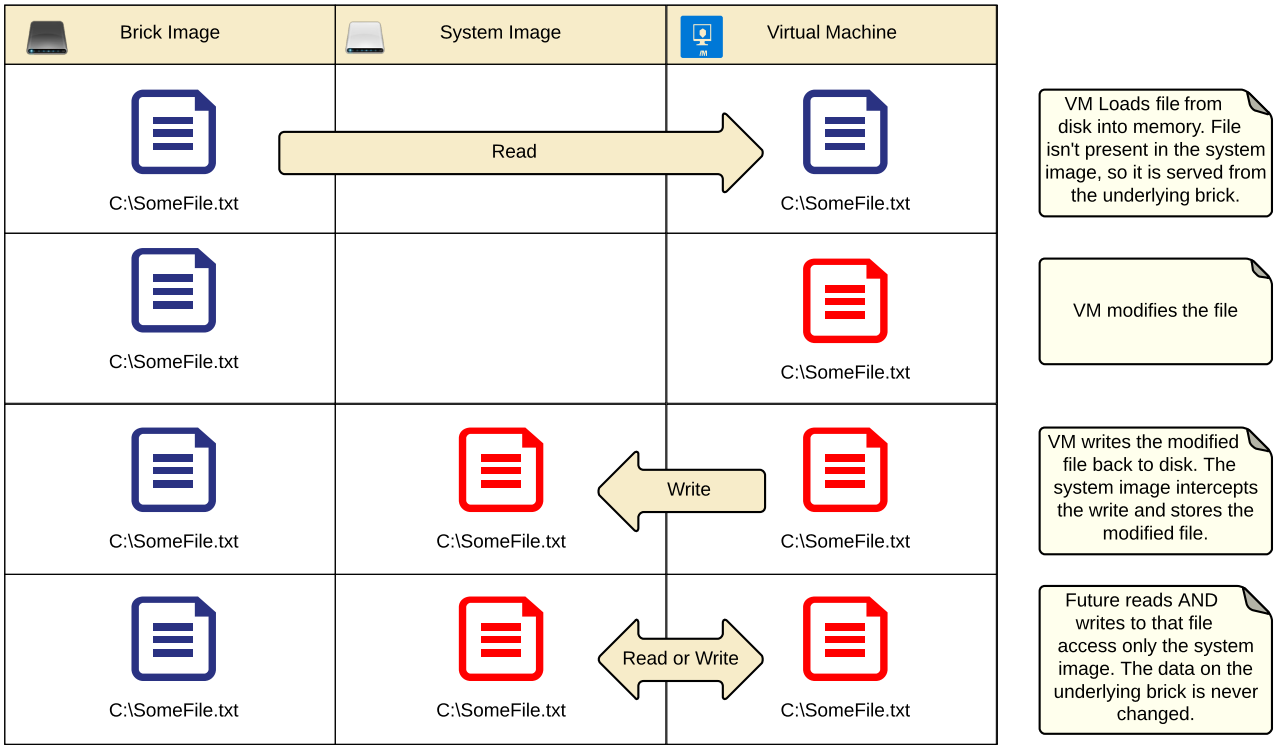

VM stays up until destroyed by the user or an administrator.
Stage 6: Shutdown and Deletion
VM
VM System Image
Brick Image
All drives are detached from the VM, and then the VM is shut down. Once the VM has shut down, the copy-on-write system image it was using is deleted.

Virtual Machine Templates ("Bricks")



Thin images on top of the brick. Contain only differences between the VM's current state and the underlying image using what is called "copy-on-write".

"Copy-on-Write" Semantics



Virtual Machine File System Structure