
SiestaTime Documentation

Release beta

Alvaro Folgado

May 26, 2021

1	Installation	3
1.1	Dependencies	3
1.2	Hive	5
1.3	Client	7
2	Adding Resources	9
2.1	Operators	9
2.2	Virtual Private Cloud	9
2.3	Domains	12
3	Hive Status	15
3.1	Jobs,Logs	15
4	Deployment	17
4.1	Implants	17
4.2	Post. Servers	22
5	Interactions	25
5.1	Implants	25
5.2	Bichitos	27
5.3	Post. Servers	32
6	Reporting	37
6.1	Basic Reports	37
7	Concepts,Bugs and Dependencies	39
7.1	Theory: Why Siesta Time ?	39
7.2	Known Bugs	41

Siesta Time is a red team framework with the purpose of automating and facilitating implants creation and their C2 Infrastructure. All these actions keeping trace of what your Operators are doing.

Save your Different Network Infrastructure Elements Save units of virtual private cloud, domains and software as a service credentials to be able to deploy C2 infrastructure.

Deploy Implants that adapt to your current target Select Implant configuration, modules and set of redirectors. The different modules available will define how the implant will egress, persist in target foothold. . . The output will be binaries for different devices and the working C2.

Maintain Access and Interact with your foothold Once the delivery process is successful and the implant is executed, list your infected devices. Interact with them to send essentials commands or put them to sleep.

Deploy Post. Exploitation Servers for Lateral Movement With the objective to achieve further target's assets from their intranet, Siesta Time can use your resource battery to deploy "shorted lived" handlers (ssh, msf, empire. . .)

Document the whole Operation to provide insights to the target institution's Blue Team or Stakeholders

Reporting will focus on saving every framework job and action execute in footholds or Post. Exploitation handlers.

CHAPTER 1

Installation

SiestaTime installation needs just two elements to work properly, the Hive, normally installed by the operations admin/manager, and the client, for the Operators.

Note: Installation for both Hive and client is designed for the moment just for Ubuntu (tested on Ubuntu 18.04.4 LTS)

Everything related to the installation process for both Hive and client will be saved on:

```
$ ./SiestaTime/installConfig
```

- **Dependencies:** *Windows* | *Darwin*
- **Hive:** *Online* | *Offline*
- **Client:** *Install*

1.1 Dependencies

Some of the modules and capabilities of Bichito are written using native libraries of target OS Devices. For the proper functioning of them, target OS dependencies need to be acquired.

1.1.1 Why Dependencies?

SiestaTime is designed to generate implants for three platforms: **Linux, Windows and Darwin**. It will use Go to compile the implants for target OS and architecture, but some of the implants functionalities require more specific dependencies. To do the “system level delicate stuff” SiestaTime will use `cgo` and `C wrappers` to call C++,

Objective-C, which will require certain target OS dependencies (that will be statically linked in the compilation process of the implants).

- Windows: CGO + MinGW
- Darwin: CGO + OSXCross

1.1.2 Get Dependencies from Windows 10

To properly install Have a set of dependencies extracted from Windows need to be in place first. This will be necessary to be able to compile some implant modules for windows. The result of this step should be a zipped folder with the following structure:

```
includes/  
  taskschd.h  
libs/  
  x86/  
    comsupp.lib  
  x64/  
    comsupp.lib
```

This zip needs to be placed:

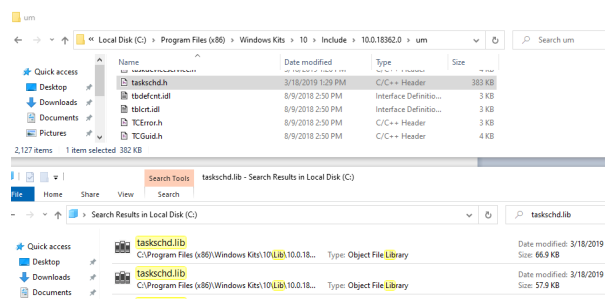
```
$ ./SiestaTime/installConfig/windependencies.zip
```

The command you can run for generating the resulting zip:

```
$ zip -r windependencies.zip *
```

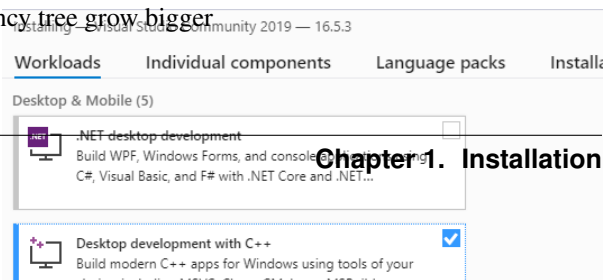
Search for target dependencies

- Perform a clean Windows 10 Installation (EG. virtual machine)
- Install microsoft visual studio community() with C++ Console apps Development



- Copy target files keeping previous folder structure. You can just find the files on windows after Visual Studio with C++ support is installed (listed at ./SiestaTime/installConfig/implant_dependencies.txt)

Note: A script will be provided in the future when dependency tree grow bigger



1.1.3 Get Dependencies from Darwin (Mac OSX)

- A. Install XCode on MacOSX 10.13
- B. `git clone https://github.com/tpoechtrager/osxcross`
- C. Run osxcross extract script

```
$ ./tools/gen_sdk_package.sh
```
- D. Copy output file to SiestaTime/installConfig/
MacOSX10.13.sdk.tar.xz

Note: Since i386 has been dropped <= most ideally 10.13 of your Darwin OS should be used

1.2 Hive

Hive is the main Operations Server of Siesta Time. Is the first element that needs to be deployed. The user has the option to do it *Offline*, or using terraform to deploy Hive with target Virtual Private Cloud resources. Configurations of target VPC to use will be saved in a txt file by the syntax of: `./SiestaTime/installconfig/config<VPC>.txt` There is the option to deploy without using a VPC, or *Offline*, this will install Hive in the current host.

1.2.1 What this does? What is Hive?

Hive is the main “Operation Server”. Hive will receive every command/job from authenticated clients and process or redirect them to a target foothold. The main DB of siesta time will be in this server, with all information/configurations from the red team operators’ actions.

- Install in the installer’s device host system dependencies (gcc,apache utils,...)
- Parse config<VPC>.txt file, use the parameters for creating a hive.tf (terraform plan)
- Download Go and the required dependencies to compile Hive.
- Create Hive sqlite DB
- Download Terraform and apply plan

In the same way, the hive.tf plan will (and this will be performed in the same host if *Offline*):

- Install Hive OS/Distro. dependencies

- Download go and their dependencies, to be able to compile Implants
- Download Terraform and terraform plugins
- Create `/usr/local/STHive` folder structure
- Upload OS dependencies, keys, sqlite DB, compiled Hive binary ...
- Configure Hive as a service

1.2.2 Online - AWS

Steps to Prepare AWS Servers

- **Find EC2 Information**

- Prepare AWS key and credentials for target VPC
- AccessKey/SecretKey
- EC2 → “My Security Credentials” → “Access Keys”
- `ami`
- `region`
- Create key pair on target region and Download “.pem” key
- **Complete** `SiestaTime/installConfig/configAWS.txt`

```
USERNAME : Admin Username
PASSWORD : Admin password
port : HTTPS Hive port listener
accesskey: AWS accesskey
secretkey: AWS secretkey
Region: AWS region
Keyname: AWS keyname (without .pem)
ami: aws ami
itype: AWS ec2 itype
```

- Copy AWS key to `SiestaTime/installConfig/<keyname>.pem`
- Run

```
$ ./hive.sh installaws
```

1.2.3 Offline

Offline option let operators to install hive in a target host without the use of terraform or any kind of VPC credentials.

```
./hive.sh installOffline <IP> <Port>
<targetFolder> <adminUsername>
<adminPassword>

$ ./hive.sh installOffline 0.0.0.0 6232 /usr/local/ admin admin
```

Note: Every installing option comes with a “No Darwin” version of it. This will let hive to work without the need of Darwin dependencies (but loosing MacOSX implant abilities)

```
$ ./hive.sh installawsNoDarwin
$ ./hive.sh installOfflineNoDarwin
$ [...]
```

1.2.4 Uninstall

```
$ ./hive.sh remove
```

Warning: When installed Offline remove will not erase created/configured host data and packages

1.3 Client

Once Hive is Online, operators can connect to it, but they need to install their client first

1.3.1 What this does? What is the Client

GUI electron powered and Go application running in Operators’ devices. The go client executable run a localhost server that feeds data to the electron app. In the same time, the go client executable authenticate against Hive with credentials passed in compiled time. Clients will authenticate against hive and send jobs to it. They can also interact with Post. Servers thanks to a Hive tunnelization system

- Install Client OS dependencies
- Use config<VPC>.txt to get user credentials
- Download Go and dependencies to compile client
- Configure and install electron app

1.3.2 Install

Run `stime.sh` for client installation. This will compile `stclient` with provided credentials and configuration, and will generate the GUI folder. `stime.sh install` `<username>` `<password>` `<Hive IP/Domain>` `<Hive Port>` `<Client Port>` `<Hive VPC certificate Fingerprint>`

```
$ ./stime.sh install admin admin 13.57.31.79 6232 8000 $(openssl x509 -fingerprint -sha256 -noout -in
```

1.3.3 Run the Client

```
$ ./stime.sh
```

1.3.4 Uninstall

Will remove go dependencies `./stime.sh remove`

CHAPTER 2

Adding Resources

Saving online infrastructure resources will be the first steps to use Siesta Time. If you don't desire to use automatic deployment of implants you can always deploy an Offline version of them where just executables will be generated.

- **Virtual Private Clouds:** [AWS](#)
- **Domains:** [GoDaddy](#) | [Gmail](#)
- **Operators:** [Operators](#)

2.1 Operators

Admin User (the one created alongside Hive) will be able to add new operators to the Hive DB. **This will let other operators on your team to log and perform any other action but add operators.**

2.1.1 Adding Operators

Operators --> Add Operator

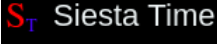
2.2 Virtual Private Cloud

This tab will be used to handle VPC instances that you can save for their late use in deployment of resources.

VPS --> Add VPS


2.2.1 AWS

- **Inputs**
 - VPS Name: Resource Name

 Siesta Time


Logged as admin

OPERATIONS SERVERS


 Hive

▼

IMPLANTS


 List Implants

▼


 Generate

▼


REGISTAR BATTERY

 VPS

▼


 Domains

▼


 SaaS

▼

STAGING SERVERS


 Droplets

▼

 Handlers


▼

REPORTING


 Basic Reports

▼

OPERATORS

 Operators

▼

 Add Operator

Implants

0

Redirectors

0

Bichitos

0

Add Operator

Username

Password

Add Operator

- VPS Type: ec2 instance type
- Access Key:
- Secret Key:
- Region: `region`
- AMI: `ami`
- SSH Keyname: EC2 keyname
- SSH Key: EC2 pem key string

The screenshot shows the Siesta Time web interface. On the left is a dark sidebar with the Siesta Time logo and a navigation menu. The main content area is light gray and displays the 'Add VPS' form. At the top of the main area, there are three counters: 'Implants' (0), 'Redirectors' (0), and 'Bichitos' (0). The 'Add VPS' form contains several input fields for configuring a new VPS.

Siesta Time
Logged as admin

OPERATIONS SERVERS

- Hive

IMPLANTS

- List Implants
- Generate

REGISTRAR BATTERY

- VPS**
 - Add VPS
 - List Registered VPS
- Domains
- SaaS

STAGING SERVERS

- Droplets
- Handlers

REPORTING

- Basic Reports

0 Implants | **0** Redirectors | **0** Bichitos

Add VPS

VPS Name

VPS Type

Access Key

Secret Key

Region

AMI

SSH Keyname

SSH Key

VPS SSH PEM Key...

Add VPS

2.2.2 Azure

TBD

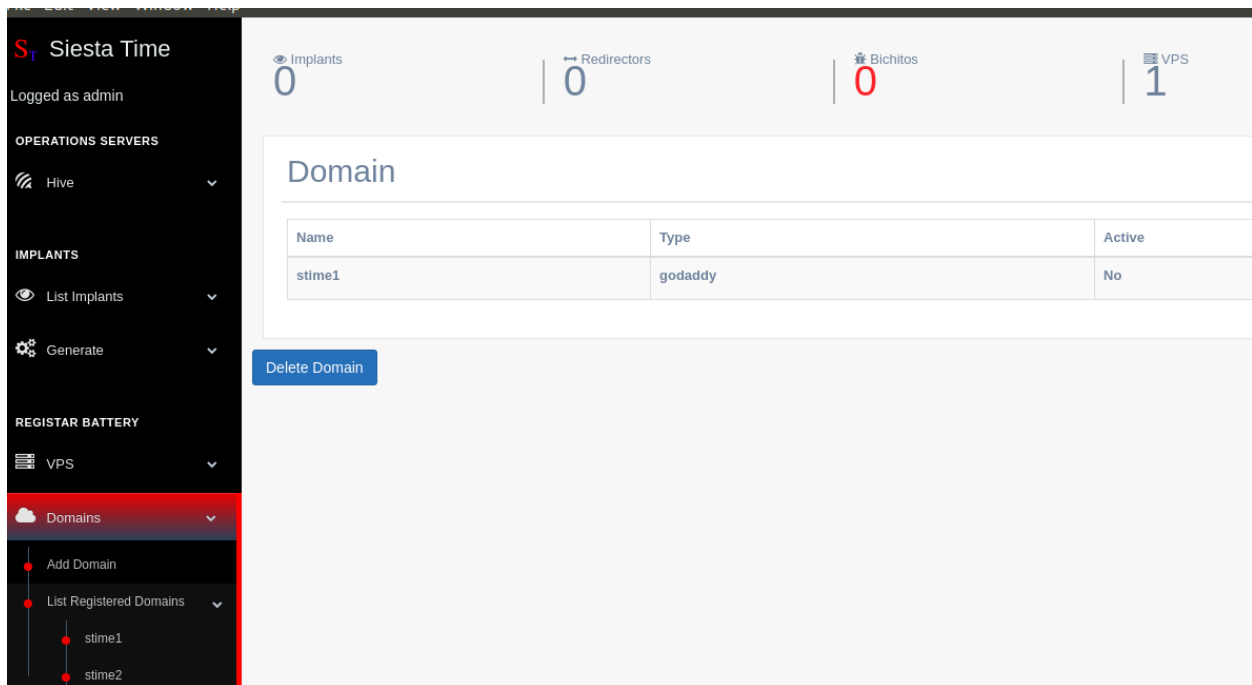
2.3 Domains

Domains let the operator to add domain resources that can be used in the C2 deployment. Operators can list them to see if they are in use by any infrastructure already with the attribute `active`

List Domain Info/Remove: `Domains/SaaS --> example.xyz`

Add New domain: `Domains/SaaS --> Add Domain`

2.3.1 Domain Status/Remove



2.3.2 Godaddy

Keys -> `godaddykeys`

2.3.3 Gmail

- **How to get my gmail connected App Credentials?**
 - Create gmail Account
 - Follow instructions -> `goquickstart`
 - Put both `credentials.json` and `token.json` Strings

Note: You need to specify the following gmail app access `google.ConfigFromJSON(b, gmail.GmailModifyScope)`

The screenshot displays the Siesta Time web application interface. On the left is a dark sidebar with the 'Siesta Time' logo and 'Logged as admin' status. The sidebar contains several expandable sections: 'OPERATIONS SERVERS' (with 'Hive'), 'IMPLANTS' (with 'List Implants' and 'Generate'), 'REGISTAR BATTERY' (with 'VPS'), 'Domains' (highlighted in red, containing 'Add Domain' and 'List Registered Domains'), and 'STAGING SERVERS'. The main content area has a top header with three counters: 'Implants' (0), 'Redirectors' (0), and 'Bichitos' (0). Below this is the 'Add Domain' form, which includes input fields for 'Domain/SaaS Name', 'Domain Type' (pre-filled with 'Go Daddy'), 'Domain' (pre-filled with 'domain.xzy...'), 'Access Key', and 'Secret Key'. A blue 'Create Domain' button is at the bottom of the form.

Siesta Time
Logged as admin

OPERATIONS SERVERS

- Hive

IMPLANTS

- List Implants
- Generate

REGISTAR BATTERY

- VPS

Domains

- Add Domain
- List Registered Domains

STAGING SERVERS

Implants: 0 | Redirectors: 0 | Bichitos: 0

Add Domain

Domain/SaaS Name


Domain Type

Domain

Access Key


Secret Key

Create Domain


 Siesta Time


Logged as admin

OPERATIONS SERVERS


 Hive


IMPLANTS


 List Implants

 Generate

REGISTAR BATTERY


 VPS


 Domains


 SaaS

Add SaaS

List Registered SaaS

 Implants
0

 Redirectors
0

 Bichitos
0

Add Domain

Domain/SaaS Name

Domain Name...

Domain Type

Gmail API

Credentials.json

Gmail Cred Json File...

Token.json

Gmail Access/Refresh Token...

Create Domain

Hive Status

The Hive Status provide red teamers a bunch of tabs where they can check every job processed by Hive. If any problem appears during the process of these, the Logs will show errors details.

- **Hive Jobs:** *Jobs*
- **Hive Logs:** *Logs*

3.1 Jobs,Logs

Hive tab will provide all information about jobs processed by Hive and errors caused alongside Hive lifespan.

3.1.1 Jobs

3.1.2 Logs

OPERATIONS SERVERS

Hive

Jobs

Logs

IMPLANTS

List Implants

Generate

REGISTAR BATTERY

VPS

Jobs

Hive

Click Job to Show Results

Cid	Jid
C-DEFAULT	J-9gAcpNlj
createVPS(VPS west1 Created)	
C-DEFAULT	J-UqRviLDK
C-DEFAULT	J-ELWXKj9n
C-DEFAULT	J-StZfXyHx
C-DEFAULT	J-NlmWGOKc

S_T Siesta Time

Logged as admin

OPERATIONS SERVERS

Hive

Jobs

Logs

IMPLANTS

List Implants

0 Implants

0 Redirectors

0 Bichitos

Logs

Hive

Time	Error
12/04/2020 21:04:15 UTC	User Auth(Bad Username/pwd):sql: no rows in result set
12/04/2020 21:04:16 UTC	User Auth(Bad Username/pwd):sql: no rows in result set
12/04/2020 21:04:16 UTC	User Auth(Bad Username/pwd):sql: no rows in result set
12/04/2020 21:04:16 UTC	User Auth(Bad Username/pwd):sql: no rows in result set

CHAPTER 4

Deployment

Once the red team has registered every resource that they want to use for their C2 infrastructure these can be used to generate it. Implants (redirectors and executables) and Post. Exploitation servers (droplets, handlers...)

- **Implants:** *Basic Config.* | *Network Modules* | *Persistence Modules* | *Redirectors*
- **Post. Servers:** *Droplets* | *Handlers*

4.1 Implants

Generate the different executables to keep connection to the foothold and their C2 (redirectors)

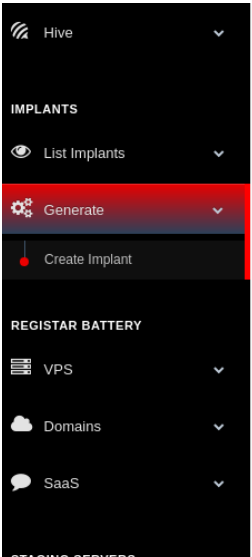
4.1.1 What is a module?

Siesta Time provides you options that define essential properties of the implant. For the moment modules can be chosen for both egression and persistence. In the future other modules could be chosen for more specific implant properties and behaviour once executed.

4.1.2 Basic Configurations

Time to Live (TTL)

Seconds to finish kill itself if the implant is not able to reach any of the redirectors.



Craft Implants

Implant Name

TTL

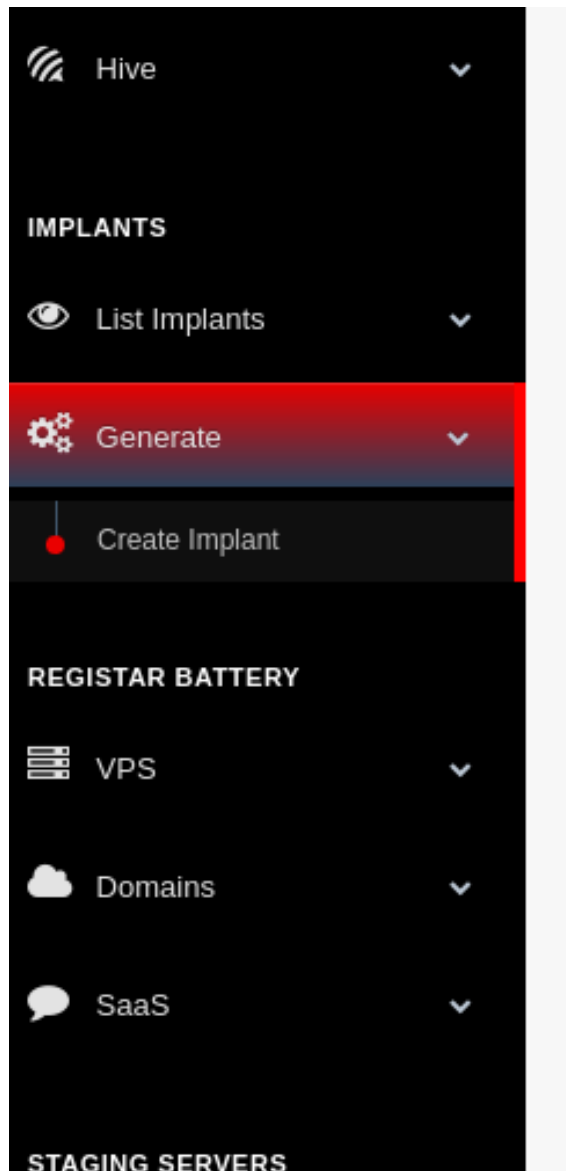
Response Time

Userland Persistence Module OSX

Userland Persistence Module Windows

Userland Persistence Module Linux

Network Module



Craft Implants

Implant Name

TTL

Response Time

Userland Persistence Module OSX

Userland Persistence Module Windows

Userland Persistence Module Linux

Network Module

Response Time

Seconds to response to a job arrived through any redirector

4.1.3 Network Modules

Network modules are platform-agnostic. It defines the communication channel between the implant and the redirectors. Each module has its own parameters.

NOTE

Network Module

Self Signed https Go

Self Signed https Go

Paranoid Self Signed Https Go

SaaS:Gmail API

SaaS:Gmail API, Mimic Https Client Fingerprints

Self Signed https Go -- No Infrastructure

Paranoid Self Signed Https Go-- No Infrastructure

SaaS:Gmail API -- No Infrastructure

SaaS:Gmail API, Mimic Https Client Fingerprints -- No Infrastructure

Note: Every network module has its No Infrastructure version which let operators define IP/Domains that are not part of the Hive battery. This will trigger the Offline mode to generate Implants. In this mode Hive will not deploy redirectors.

Self-Signed HTTPS Go

Https Client: Go Redirector Certificate: Self Signed

TLS Port → Choose the Redirector https listening port

Paranoid Self Signed HTTPS Go

Https Client: Go Redirector Certificate: Self Signed The implant will check target redirector fingerprint

TLS Port → Choose the Redirector https listening port

Warning: This module will avoid the implant to egress if there is a https TLS proxy on the middle

SaaS: Gmail API

Https Client: Go Redirector Certificate: Google Servers

SaaS: Gmail API,Mimic Https Client Fingerprints

Https Client: Mimic target TLS Fingerprint Redirector Certificate: Google Servers

UserAgent → Choose the client User Agent TLS JA3C Fingerprint → Choose the Redirector TLS Fingerprint

Warning: This module will try to evade most of NIDS non DPI (deep packet inspection) based

4.1.4 Persistence Modules

Once the red team achieves the execution of the implant through a delivery method the most important next step after checking C2 connectivity (or even before) will be to persist. This will let the implant to re-execute itself after any device shutdown/re-login

Windows - schtasks

Use windows C++ `comsupp.lib` and `taskschd.lib` to create a task that runs on user login

Userland Persistence Module Windows

Schtasks_Native

Schtask Name

Name...

Implant Path (Relative to Default User Home Folder)

folder\folder\filename...

Name → task name that will appear on windows task list Path → Windows path for target Implant Binary (`$(UserHOME)\\"folder1\folder2\execuablenameand.extension)`)

Darwin - launchd

Using Go file libraries, it writes a new launchd service. MacOSX will fetch it on user reboot/login and load the implant.

Userland Persistence Module OSX

Launchd_Plist

Launchd Name (~/.Library/LaunchAgents/com.name.agent.plist)

Name...

Implant Path (Relative to Default User Home Folder)

folder/folder/filename...

Name → Launchd file name that will appear on windows task list Path → Darwin path for target Implant Binary (\$(UserHOME) / "folder1/folder2/execuablenameand.extension")

Linux - XDG

User Desktop Linux devices with graphical interface respect some specification that let users to configure default tasks on user login. Files are written using default GO file libraries.

Userland Persistence Module Linux

Autostart_XDG

Autostart File Name (\$XDG_CONFIG_HOME/.config/autostart/name.desktop)

Name...

Implant Path (Relative to Default User Home Folder)

folder/folder/filename...

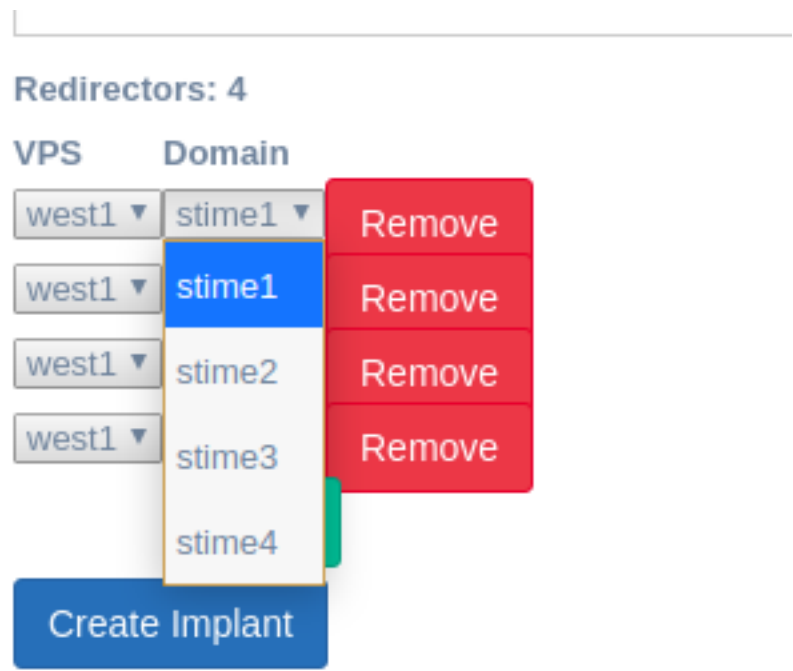
Name → Autostart file name that will appear on windows task list Path → Linux path for target Implant Binary (\$(UserHOME) / "folder1/folder2/execuablenameand.extension")

Note: Extra doc: [xdg](#)

4.1.5 Redirectors

C2 Servers where implants try to connect to retrieve commands from Hive. Implants will try one by one to connect to them, once they find out which one the can reach by the previous network methods.

Online



Warning: Domains that are active already will not appear on the list

Offline

4.2 Post. Servers

4.2.1 Droplet

A plain ubuntu Server used to drop created Implants
Name → Just the name of the resource VPS/Domain → Choose one from the battery
Https Port Path for Implant folder → /var/www/"path"/implant Endpoint: https://domain/implantpath/implant

4.2.2 Reverse SSH

The reverse SSH will create a ubuntu Server with a sshd connection and a "anonymous" user configured with its own keys. This user is configured without a bash shell, the

← → ↻

stime2.xyz/implantpath/

Index of /implantpath

Name	Last modified	Size	Description
Parent Directory	-		

Chapter 4. Deployment

Apache/2.4.29 (Ubuntu) Server at stime2.xyz Port 443

Redirectors: 2

Domain

domain1.com

Remove

8.22.3.4

Remove

Add

Create Implant

idea is that the implant will connect with that anonymous user and serve its own SSHD Server from Golang Code.

In this way, what it looks from the foothold as a SSH outbound connection, will be a remote bash/cmd serving.

[More details in developer Guide]

Warning: For the moment these SSH are not fully interactive

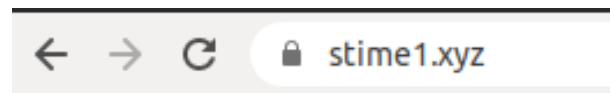
4.2.3 Reverse RDESKTOP

TBD

4.2.4 Empire, Metasploit

Similarly to the reverse SSH server, Empire and MSFT will create remote handlers to receive incoming shells.

The handler is configured with https self signed certificates. You can choose the handler port



It works!

The screenshot displays the Siesta Time web application interface. On the left is a dark sidebar with the logo 'S_T Siesta Time' and the text 'Logged as admin'. The sidebar contains several menu sections: 'OPERATIONS SERVERS' with a 'Hive' item; 'IMPLANTS' with 'List Implants' and 'Generate' items; 'REGISTAR BATTERY' with 'VPS', 'Domains', and 'SaaS' items; and 'STAGING SERVERS' with 'Droplets', 'Create Droplet', and 'List Droplets' items. The 'Droplets' item is highlighted with a red background. The main content area has a light gray header with 'Implants 0' and 'Redirectors 0'. Below this is a 'Generate Staging' form with the following fields: 'Staging Name' (text input), 'Staging Type' (dropdown menu showing 'HTTPS (Let's Encrypt) Droplet'), 'VPS' (dropdown menu showing 'west1') and 'Domain' (dropdown menu showing 'stime1'), 'Droplet HTTPs Port' (text input showing '1244'), and 'Path for Implants' (text input showing 'namepath'). A blue 'Create Staging' button is at the bottom of the form.

S_T Siesta Time
Logged as admin

OPERATIONS SERVERS

- Hive

IMPLANTS

- List Implants
- Generate

REGISTAR BATTERY

- VPS
- Domains
- SaaS

STAGING SERVERS

- Droplets**
- Create Droplet
- List Droplets

Implants 0 | Redirectors 0

Generate Staging

Staging Name

Staging Type

VPS **Domain**

west1 ▼ stime1 ▼

Droplet HTTPs Port

Path for Implants

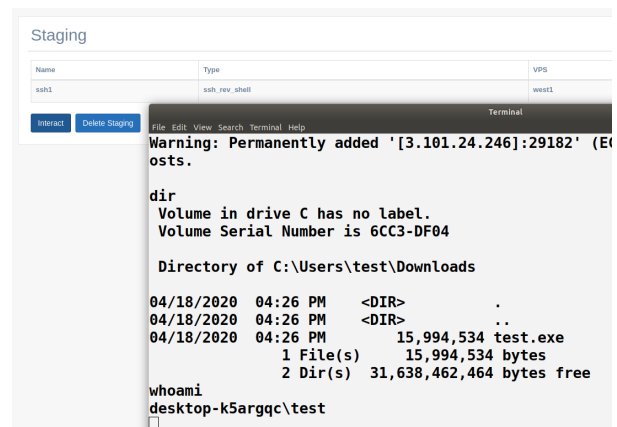
Create Staging

After the deployment the red team should design a kill-chain/attack that requires normally the delivery of the previous generated executable. Once the delivery is successful, Hive will start to receive connections from the foothold. Lateral movement may be required to acquire further target assets. Bichito has the capability to retrieve basic commands that will be used to persists or escalate privileges. If the operator needs a more complex connection to the foothold (ssh,remote desktop, tunnelization...) Post. Exploitation servers will be used. through new processes triggered from Bichito.

- **Implants:** *Implant Info* | *Download*
- **Bichitos:** *Status* | *Console*
- **Post. Exploitation:** *SSH* | *MSF* | *Empire*

5.1 Implants

Once the Implants and Post. Servers are created and you already have the infrastructure ready to go on a Red team operation. Now you are ready to download them for delivery purposes.



5.1.1 Information

```
Implants --> ImplantName
```

5.1.2 Download Implant

Once the implant is listed, it is possible to download from hive both implant and redirector executables

```
Implants --> ImplantName --> Download
```

S_T Siesta Time
Logged as admin
OPERATIONS SERVERS
Hive
IMPLANTS
List Implants
test1

1 Implants

1 Redirectors

0 Bichitos

1 VPS

Implant test1

Network	Persistence	Domains/SaaS
selfsignedhttpsgo	None	west1

Downloads test1

Target Operating System

Linux

Target Device Architecture

x32

Download Implant

Download Redirector

Note: Every Download on Hive will be made to SiestaTime/installConfig/downloads

5.1.3 Delivery - Attacks

Once the implant is created some Delivery options or “Attacks” will be available. In the future, delivery options like phishing, macro... will be available.

Drop Implant

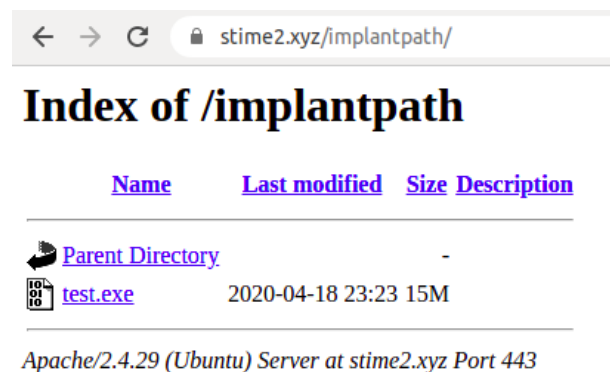
To drop implant simply choose a previously created droplet

HTA

TBD

5.2 Bichitos

Once the delivery of the implant (phishing, lv 2 mitm on the network, physical access...) is completed, Hive will start to receive Online Bichitos.



5.2.1 List Bichitos, Information

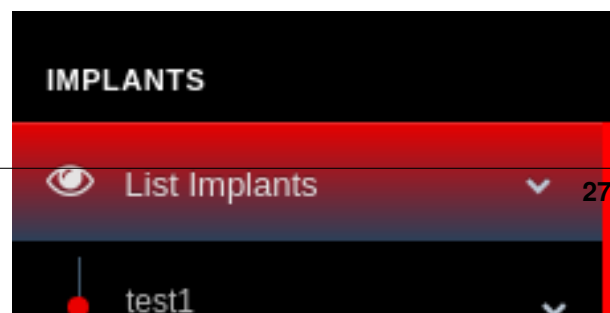
The active footholds will appear under the Implant itself. They will be first organized by ***Hostname***. Within each Hostname, Offline footholds (they were online and become offline) Under Offline active Footholds will be ready to interact with.

Note: A Offline bichito or foothold, will be determined if a job is sent, and no response back from any redirector is obtained in respTime **[more details on the developers' guide]***

- **Params.**

- ID → Target IMplant process ID

5.2. Bichitos



Bichito/Device Information

B-xlbHxbc8

ID	PID	Arch	Op. System	OS Version
B-xlbHxbc8	7980	Compiled for amd64: x64 (AMD or Intel)	windows	CurrentVersion 18363

MAC Addr.	User Name	Admin/ROOT?
00:0c:29:61:16:59	DESKTOP-K5ARGQC\test	Integrity level:8192

- PID → Target foothold PID for bichito
- Architecture → target hardware process arch
- Mac Addr
- OS Version
- Hostname
- UserName → the user context that the bichito is executed with
- AdminRoot → will show how privileged the process is in the target OS
- Redirector attached → Last redirector where the foothold connected to
- Status → Offline if no response within responseTime

5.2.2 Jobs

Similarly to Hive Jobs, this log will show jobs that are being sent to be processed by target foothold

5.2.3 Logs

Error log generated by the foothold, for example, when it is unable to connect to one redirector

5.2.4 Interact - Job Console

This is the main Job-based console of Siesta Time Jobs are sent to the Hive, then the Hive detects which redirector the foothold is attached to, and finally the foothold grabs the Job and response back in `respTime` Results will be shown in the Bichito's Job Log



clear

Clear JS console

respTime <seconds>

Change target Bichito respTime to target integer in seconds

ttl <seconds>

Change the time to live of the target Bichito to target integer seconds

exec <commandString>

Pass target string to default command interpreter (sh/bash/cmd) and retrieve output

EG.: `exec ipconfig`

ls <pathFolder>

List target foothold Folder

EG.: `ls C:\\`

C-DEFAULT	J-ziylO0YX	12/04/2020 18:39:48 PDT
<div>Windows IP Configuration</div> <div>Ethernet adapter Ethernet0:</div> <div>Connection-specific DNS Suffix . : localdomain</div> <div>Link-local IPv6 Address : fe80::3550:69db:8f85:8c17%9</div> <div>IPv4 Address. : 172.16.37.142</div> <div>Subnet Mask : 255.255.255.0</div> <div>Default Gateway : 172.16.37.2</div>		

Jobs B-pwCv8BTZ

Click Job to Show Results

Cid	Jid	T
C-DEFAULT	J-NmWnpWSD	1

<div>-----Directories-----</div> <div>\$Recycle.Bin</div> <div>PerfLogs</div> <div>Program Files</div> <div>Program Files (x86)</div> <div>ProgramData</div> <div>Recovery</div> <div>System Volume Information</div> <div>Users</div> <div>Windows</div> <div>-----Files-----</div> <div>Documents and Settings</div> <div>pagefile.sys</div> <div>swapfile.sys</div>
--

Jobs B-I9PNrUMG

Click Job to Show Results

Cid	Jid	Time	Job
C-DEFAULT	J-Zp575NEK	18/04/2020 16:28:02 PDT	accesschk

File name Bytes Permissions SID Last Modified
test.exe 15994534 -rw-rw-rw- S-1-5-21-3733192527-2205489211-2205116887-1001 2020-04-18 16:26:21.5962542 -0700 PDT

accesschk <pathFile>

Provide target operating system's file access control rules as an output

EG.: accesschk C:\\windows\\system32\\calc.exe

read <pathfile>

Read bytes of target file and retrieve the output

write <string> <pathfile>

Write string on target File. If a file exists, it will append at the end.

wipe <pathfile>

Wipe target file. Will transform its bytes to 0 and then remove the disk header.

upload <operator-PathFile> <Foothold-Pathfile>

Upload Operators source file to target foothold location

EG.: upload /home/rebujacker/upload1 C:\\Users\\test\\Desktop\\

download <Foothold-pathfile> <operator-pathfile>

Download target foothold file to a operator folder

EG.: download C:\\Users\\test\\Desktop\\download1 /home/rebujacker/download1

Note: Every Output is set to hold a maximum of 20 Megabytes. For moving bigger blobs of data you need to use POST./Staging Servers

kill

Kill actual bichito process

removeInfection

If the implant was configured to hold a persistence on target foothold, this command will:

- **Steps**
 - Remove/Wipe Persistence configurations/files
 - Remove/Wipe executable on disk, if applicable.
 - Kill the actual bichito process

Note: This command output will be always “Processing” since the target foothold will never answer back after removal

5.2.5 Attach Bichitos/Footholds to Post. Servers

Once the foothold is running one of your implant processes (Bichitos) the natural way to interact for non basic operations will be the Post./Staging Servers that can be created/deleted at any moment. For the moment, these attacks/injections come in the following way:

SSH Rev Shell

Create a new thread from the actual Bichito process. This thread will create an outbound SSH client request against a target previously created “Rev SSH Handler”

Empire

Generate a launcher that will be executed by a bichito thread using “Exec” and “python”. Python needs to be installed on target foothold previously.

Metasploit

TBD

Note: These attack/Injections normally have a Offline option, that will let Operators to build their own Post. Servers without hive

Warning: For the moment these actions are not creating new processes, or injecting in other’s processes, that is TBD

Attacks B-I9PNrUMG

Inject Shell

SSH Rev Shell

Staging Server

ssh1

Inject

Job Log

Error Log

Interact

5.3 Post. Servers

Most of the previously created Handlers can be interacted with. This menu will be found on “Handlers” tab

5.3.1 Interact with SSH - Full Interactive Shell

Interact with a SSH Rev Handler will be possible once a bichito has been attached to it.

Staging

Name	Type	VPS
ssh1	ssh_rev_shell	west1

[Interact](#) [Delete Staging](#)

Terminal

```
Warning: Permanently added '[3.101.24.246]:29182' (ECDSA host key).

dir
Volume in drive C has no label.
Volume Serial Number is 6CC3-DF04

Directory of C:\Users\test\Downloads

04/18/2020  04:26 PM    <DIR>          .
04/18/2020  04:26 PM    <DIR>          ..
04/18/2020  04:26 PM                15,994,534 test.exe
               1 File(s)      15,994,534 bytes
               2 Dir(s)  31,638,462,464 bytes free

whoami
desktop-k5argqc\test
```

Note: The process and connection will finish once you close the Interactive shell

Warning: For the moment these ssh are full Interactive for Linux and OSX (no windows) TBD

5.3.2 Interact with SSH opening a SOCKS5

Interact with a SSH Rev Handler will be possible once a bichito has been attached to it. This will open a SOCKS5 in the Operator's machine ready to proxy all traffict through implanted device.

Warning: You will need to kill SSH sessions in the target staging server to avoid extra process/threads to be running in the implanted device

5.3.3 Kill SSH Sessions

Interact with a targer Post/Staging Server and kill any connected POST SESSIONS

5.3.4 Empire and Metasploit

Interacting with both MSFT and Empire is very straight-forward. Once created just click "Interact" and you will be provided by a console directly bonded to the target's MSFT/Empire Server.

Staging

Name	Type
empire1	https_empire_letsencrypt

[Interact](#) [Delete Staging](#)

File Edit View Search Terminal Help

Terminal

```
[ - ] Timed out waiting for child stop.  
(Empire: agents) > list  
list  
[!] No agents currently registered  
(Empire: agents) > 
```


Reports are a way to extract every action performed by Operators using the SiestaTime framework.

6.1 Basic Reports

For the moment is just a plain text document, but in the future a more elaborated report will be developed.

- **The reports will have the following strings:**

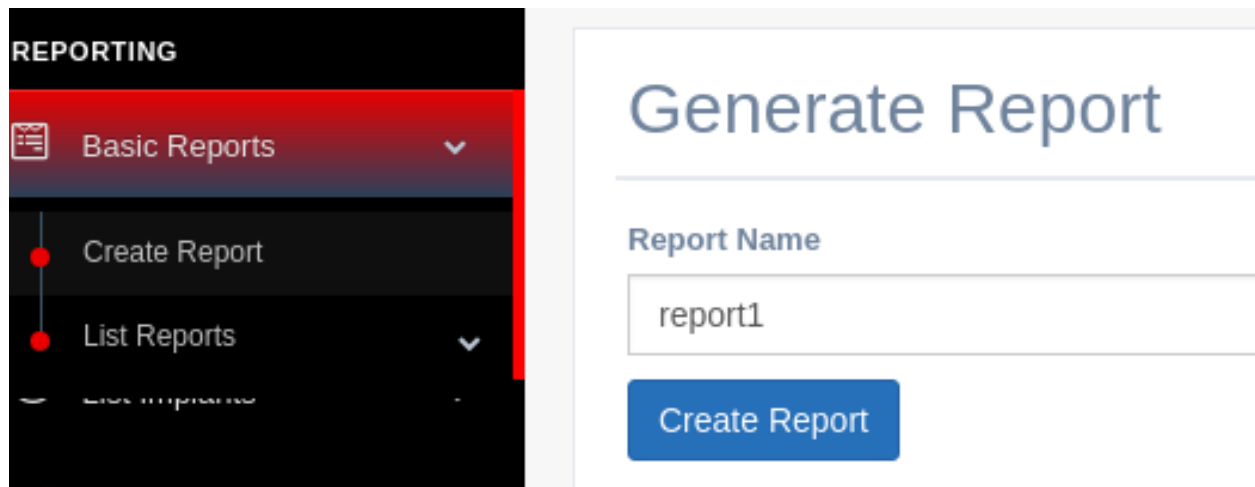
- Report Name
- Report Creation Time
- Every Job Processed by Hive
- Every Job sent to bichitos
- Every command processed through post-exploitation servers (Interactive Sessions per user and time, with their outputs)

6.1.1 Create Report

Basic Reports --> Create Report

6.1.2 List Reports,Download

Once the report is created it can be downloaded from hive. They will be automatically placed in: `SiestaTime/installConfig/reports`



Concepts, Bugs and Dependencies

SiestaTime is still in its Beta version, a lot of work and testing is still needed, these are common issues that you may encounter. Software compatibility List.

- **Theory:** *Concepts*
- **Known Bugs:** *Bugs*

7.1 Theory: Why Siesta Time ?

7.1.1 Concepts

Hive

Is the main “Operation Server”. Hive will receive every command/job from authenticated clients and process or redirect them to a target foothold. The main DB of siesta time will be in this server, with all information/configurations from the red team operators’ actions.

Operator

Is the equivalent of the user in Siesta Time. The creator of hive or “Admin” (first user) will be the only one able to add new Operators. Operators added will be able to compile their own client and connect to Hive.

VPS/VPC

“Virtual Private Service/Cloud” are sets of credentials saved in Hive that can be used to deploy redirectors that back-bone implants’ connection and Post./Staging Servers to interact with them later on.

Domain

Hive will be able to store a set of credentials to manipulate a target domain at will. Used to map its resolution to the generated Server's infrastructure selected VPCs. Once an element is requested to be created (implant,post. server...)

SaaS

“Software as a Service” are sets of credentials from an internet service that let implants to egress using string data

Implant

Implants are composed by a number of redirectors and the compiled executables for different platforms (linux,darwin and windows). The implants will have an array of redirectors to connect to, that will be in the shape of a target IP,domain or SaaS account. These redirectors will be deployed in the creation of the implant

Redirector

Host with a server software running as a service. Its purpose is to redirect jobs from footholds (bichitos) to the Hive, and vice versa.They are automatically deployed on implant creation. In offline mode, the redirector executable can be downloaded to be installed in any desired host

Bichito

The main implant of Siesta Time. They are generated in the shape of an executable for a target platform. Once executed, they will appear as an online process within the created implant and attached to a redirector. On the implant creation it is possible to choose the capabilities of the Bichitos. How will egress through the network, his time to day, persistence... these are the modules

Staging/Post

Servers whose objective is performing delivery, staging and post-exploitation tasks. Operators can directly connect/interact with them.

Report

These elements are a plain text file that holds every Job processed by Hive, and every command typed on Staging/Posts servers

Client

GUI on electron and go application running in operators devices. Clients will authenticate against hive and send jobs to it. They can also interact with Post. Servers thanks to a Hive tunnelization.

7.2 Known Bugs

7.2.1 Client

Job Creation - Stuck

The client software implements a lock to avoid the triggering of too many jobs against Hive. This still can be faulty and the lock stuck to 0. If this happens a restart on the GUI/Client is recommended.

7.2.2 Hive

Jobs Queue - GUI

Hive will execute jobs 1 by 1. Once an output for each Job is reached, the Jobs will update. This means that while Hive is busy operators will not see any updates on the client for sent jobs.

DB Locked

Multiple writes on Hive DB have been shown to block DB and miss writes. This normally happens when multiple “Hive Jobs” are sequenced too simultaneously.

7.2.3 Redirector

Not common known problems are still known in redirector software Deadlock is known to have happened in the past, but redirectors auto-restart if this happens.

Note: Having a bunch of redirectors is recommended if some of them fails

7.2.4 Bichito

Not common known problems are still known in implant/Bichito software

Note: Persistence is recommended in the scenario where the bichito/implant software could block its functionality on unknown bugs

7.2.5 Post. Servers

Note: Faulty Staging/Post. Can be easily trashed and re-deployed. If the user feels any of them is becoming faulty, use the rapid deployment properties of STime to create it again.
