3. Data Model

Navigation

Cobalt Strike's team server stores your hosts, services, credentials, and other information. It also broadcasts this information and makes it available to all clients.

Data API

Use the &data_query (functions.html#data_query) function to query Cobalt Strike's data model. This function has access to all state and information maintained by the Cobalt Strike client. Use &data_keys (functions.html#data_keys) to get a list of the different pieces of data you may query. This example queries all data in Cobalt Strike's data model and exports it to a text file:

```
command export {
        local('$handle $model $row $entry $index');
        $handle = openf(">export.txt");

        foreach $model (data_keys()) {
                println($handle, "== $model ==");
                println(data_query($model));
        }

        closef($handle);
}
```

Cobalt Strike provides several functions that make it more intuitive to work with the data model.

Model	Function	Description
applications	&applications (functions.html#applications)	System Profiler Results [**View -> Applications**]
archives	&archives (functions.html#archives)	Engagement events/activities
beacons	&beacons (functions.html#beacons)	Active beacons
credentials	&credentials (functions.html#credentials)	Usernames, passwords, etc.
downloads	&downloads (functions.html#downloads)	Downloaded files
keystrokes	&keystrokes (functions.html#keystrokes)	Keystrokes received by Beacon
screenshots	&screenshots (functions.html#screenshots)	Screenshots captured by Beacon
services	&services (functions.html#services)	Services and service information
sites	&sites (functions.html#sites)	Assets hosted by Cobalt Strike
socks	&pivots (functions.html#pivots)	SOCKS proxy servers and port forwards

targets	&targets (functions.html#targets)	Hosts and host information

These functions return an array with one row for each entry in the data model. Each entry is a dictionary with different key/value pairs that describe the entry.

The best way to understand the data model is to explore it through the Aggressor Script console. Go to **View -> Script Console** and use the `x` command to evaluate an expression. For example:

Querying Data from the Aggressor Script console

Use `on DATA_KEY` to subscribe to changes to a specific data model.

```
on keystrokes {
        println("I have new keystrokes: $1");
}
```