CyberTOOLBELT

RESTful API

Documentation

Revision:	1.0	9/3/2014
	1.1	9/27/2014
	1.2	10/16/2014
	1.3	10/27/2014
	1.4	11/6/2014
	1.5	2/11/2015
	1.7	3/18/2015
	2.0	6/19/2015
	2.1	7/22/2015
	3.0	8/17/2015
	3.1	9/15/2015
	3.2	10/26/2015
	3.3	11/29/2015
	3.4	1/8/2016
	3.5	2/15/2016
	3.5a	2/22/2016
	3.6	3/15/2016
	4.0	7/1/2017
	4.0.a	10/22/2017
	4.1	2/8/2018
	4.2	6/1/2018
	4.2a	6/21/2018

CONTENTS

INTRODUCTION

This document describes the use of RESTful API to the CyberTOOLBELT database. It is assumed that the reader has familiarity with RESTful API usage in general

This document describes version 4.x of the CTB API.

CONVENTIONS

Anything that appears in mono-spaced text with a gray background is either a data description or program code. Example:

```
<?php
//
//    This is an example bit of code
//
```

OVERVIEW

The CTB API listens on port 2476 of "https://rest-api.cybertoolbelt.com:2476/api/v2/". For PHP programmers we supply an interface class named: int_ctb.php. It is recommended that you use this class to communicate with the API. If you are using another programming language you will need to create a similar function for your programs. We have included the source code of the PHP class in Appendix C of this document. The program file itself is available upon request.

We apologize in advance for the output results from the API that appear in a couple of different formats. The API grew organically. To be a little fair, a lot of the code had been developed for various projects over the years before they were collected into the CTB API. All new development, for quite some time, returns JSON formatted results. Version 4.0 was a big step in this direction. The primary exception to the JSON preference rule is that when large amounts of data can be returned that are single or dual fields are typically returned as a string for performance reasons. Example: Domain names and their associated ID numbers.

The commands between version 3.x and 4.x are not compatible. The results and calling format in almost all cases has changed. See the separate migration document *CTB RESTful API: Migration from 3.x to 4.x* for details.

You will receive an HTTP status code of 400 if you pass a malformed request. This can be for several reasons but the most common is a missing required parameter. The single returned field "error", if returned, will detail the actual error.

If you receive a status code of 500 as the result of a call that means that there has been sort of system or software error in the API. Notify CTB support as soon as possible.

BTW: The most common cause of the 404 error is forgetting the trailing "/" in the URL.

USING THE API

The API provides access to CyberTOOLBELT's data. The data provided and the methods of access are outlined in this section. The API supports various data queries on different datasets.

The API currently only supports POST calls. The various endpoints of the API are documented in the individual dataset descriptions.

Substitute your actual CTB-issued API Key for the term API_KEY in the following documentation as part of the URL's.

Errors are returned in one of two ways as follows:

1. The HTTP status code. 200 or 0 is success. Anything starting with a 4 or a 5 is a problem.
2. The response will contain a non-blank error field as in:
   `{"error": "Daily quota limit exceeded"}`

Most errors will return a HTTP status and an error field. For example, an invalid API key will return status "403 – Not Authorized" as well as `{"error": "Not Authorized"}`.

Some of the calls will also return a warning field in the JSON. The format will be something like this: `{"warning": "domain not found"}`. These types of warnings are usually given when the data requested in not in our database.

SUBSCRIPTION INFORMATION

CTB API access is fee based. There are a number of toolsets (left column) which all require individual subscriptions in addition to the base API access. Pricing is available from CTB or our resellers. Note that a number of the toolsets contain more than one tool. For example, the Whois toolset contains three individual tools.

Toolset	**Description**	**Includes Tools (and others)**
Whois	Provides access to the Historic Whois Dataset	• Search by Whois ID • Search by domain name • Translate Whois IDs to domain names and IDs
IP Whois	Provides access to Historic IP address Whois Dataset	• Search by IP address
Domain Whois data mining	Provides ability to mine the Whois Dataset using powerful full text queries Also known as Reverse Whois.	• Whois Data Mining
IP Operations	Various IP address operations	• Reverse IP lookup • ASN Lookup by IP address • TOR node check • IP Type (proxy/Tor/VPN) • Etc
Domain Operations	Various domain-related operations	• Domain Zone File History • Find Domain Names • Find Subdomains • Domains Related to Email • Etc/
Issues	Domain/IP issues dataset	• Domain Issues • IP Issues
IP Whois data mining	Provides ability to mine IP dataset using powerful full-text queries.	• IP Whois Data Mining
Update Operations	Reporting and updating database functions	• Reporting
New Domain Notifications	Provides either a push or pull service to notify you of new domains that CTB discovers	• New Domain

USING THE PHP INTERFACE

There is a file named int_ctb.php which provides access to the API for PHP programs via the CTB class. Appendix C contains the version of the source code at the time of this printing.

The interface is used as follows:

```
include(“int_ctb.php”);
$dm=new CTB();

$status=$dm->restCommand(“ip/domains/”,array(
      “key”=> “YOUR_CTB_KEY”,
      “ip”=>”64.34.164.179”
    ));
if ($status) {
       print “Error: “.$dm->errorMsg($status).\n”;
       exit;
}

$results=$dm->GetResults();
print “$results\n”;
```

The following methods are available:

setPort(port)

This method is used to determine the port to speak to the API on. It defaults to 2476. This method is primarily used by the API developers to communicate with another copy of the API that is undergoing testing and development. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$dm->setPort(4106);
```

```
include(“int_ctb.php”);
$dm=new CTB();
$errno = $dm->getError()
```

```
include(“int_ctb.php”);
$dm=new CTB();
$errmsg = $dm->errorMsg()
```

This method is used to execute API commands. Refer to the example at the start of this section.

The first parameter passed to the call is the endpoint of the API to access. In our example, this is "ip/domains/" (don't forget the trailing "/" on all endpoints). This call returns all domains associated with the IP address passed in the parameters.

The second parameter is an associative array of key/value pairs of which, at least, "key=YOUR_CTB_KEY" must be defined. All API calls require this parameter.

getResults()

This method is used to return the results (if any) of an API call. It can also return an error/warning message if the returned status is not zero.

API COMMANDS

This section details the various API endpoints (also referred to as calls or commands). They are grouped according to their primary endpoint.

Results of calls are mostly returned as JSON. In some cases, there is only a single field returned: *results*. This is determined by the amount of data and the need to reduce extraneous character overhead.

Note that the individual call descriptions leave out that a required parameter is your API key. This is a required input parameter for all calls.

In the endpoint write-up, the common part of the URL (https://rest-api.cybertoolbelt.com:2476/api/v2") is not shown. In your actual calls it will preceed the listed endpoint. Our php interface automatically prepends the endpoint with the common part of the URL so that you only have to supply the endpoint.

Each write-up lists the toolset that the endpoint belongs to. You must subscribe to the toolset that is listed to be able to use the call.

Tech tip: If you are getting 404 errors on your POST calls and can't figure it out – ensure that there is a trailing "/" character on your POST calls.

DOMAIN COMMANDS

All the commands in this section are domain operations. They begin with the "/domain/" endpoint.

Search for Domains

Endpoint: domain/match/ Domain Operations Toolset

Returns domain IDs and domain names that match the passed parameters. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“domain/match/”, array(
          “key”=> “YOUR_CTB_KEY”,
           “terms”=>”xelent”,
          ”since”=>0
));
If ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

This call a JSON encoded string with a single field which is named *results* as follows:

```
[results] => 262167832:24-exelent-medication.info,262167839:24-exelent-pill.info,262167843:24-
exelent-pills.info,262167845:24-exelent-tabs.info,262167848:24-exelentmedication.info,262167850:24-
exelentpil.info,240446168:24-exelentpills.info,262167861:24-exelenttabs.info,262188347:24exelent-
medication.info,262188350:24exelent-pill.info,240446171:24exelent-pills.info,262188354:24exelent-
tabs.info,262434138:24exelentmedication.info,262434145:24exelentpill.info,262188361:24exelentpills.in
fo,262434154:24exelenttabs.info,186337438:a-xelent.de,253028731:aion-
exelent.ru,323880948:appxelent.com,216316950:axelent.be,126246082:axelent.biz,221583155:axelent.co.uk
,103461079:axelent.com,218938366:axelent.cz,186337439:axelent.de,143621358:axelent.es,144576704:axele
nt.eu,226245506:axelent.fi,147030408:axelent.fr,35431048:axelent.info,82315103:axelent.net,151246345:
axelent.nl,250487595:axelent.nu,36172196:axelent.org,293110056:axelent.pl,98390803:axelent.pro,154277
614:axelent.ru,237555917:axelent.se,235315470:axelent.sk,313658778:axelent.us,136171584:axelentbelgiu
m.be,308099192:axelentbenelux.com,209221277:axelente.com,88619215:axelentengineering.com,159134152:ax
elentengineering.se,213830973:axelentermedia.co.jp,128114203:axelentermedia.com,150038703:axelenterme
dia.jp,57626985:axelentermedia.net,47098748:axelenterprise.com,289996210:axelenterprises.com,46481625
:axelenterprises.net,39939034:axelentertainment.com,313567287:axelentgroup.com,354404811:axelentpartn
ers.com,111521378:axelentracing.com,102086026:axelentsafety.com,186337440:axelentsafety.de,255690192:
axelentsafetystore.com,310489094:axelentsoftware.com,237613093:axelentsro.sk,65684295:axelenttecnel.c
om,62440927:axelentusa.com”
```

The returned information consists of zero or more comma-separated entries. Each entry represents a single matching domain. There are two pieces of data in each entry separated by colon characters. The first piece is the internal ID of the domain. The second piece is the actual matching domain name.

You pass two parameters: TERMS which contain the term(s) that the domain names must match; and, SINCE which is an-UNIX-formatted timestamp (seconds since 1/1/1970).

Note that the current time is represented in PHP as the TIME() function. So, in PHP, if you wanted to limit the search to only those domains acquired in the last 6 months, SINCE would be computed as:

```
$since=time()-(182.5*86400);
```

If SINCE is equal to 0 (as in the example shown above) then all matching domains are returned.

The search terms are either a full domain name with a TLD (ie: example.com) or a partial name without a dot to find all domains containing the partial name (ie: herbal), or you can enter the characters that a domain must start with followed by a "%" character to get all domains back that start with the characters before the "%" character. For example, the command:

```
/domain/match/
terms=pixel%
since=0
```

Will return all domains starting with the substring "pixel" without time restriction.

You can also specify AND and NOT functions as follows:

```
/domain/match/
terms=xelent +income
since=864000
```

This command will return all domains with *xelent* anywhere within the name and the substring *income* appearing anywhere in the domain name and anytime in the last 10 days.

```
/domain/match/
terms=xelent -ment
since=0
```

This command will return all domains with string *xelent* anywhere within the name but NOT also containing the substring *ment* with no time restriction.

You can string more than one condition together as in:

```
/domain/match/
terms=xelent +income -ment
since=0
```

This command returns all matches that contain the substrings *xelent* and *income* but do not contain the substring *ment* with no time restriction.

```
/domain/match/
terms=xelent +pixel +enter
since=172800
```

This command returns all matches that contain the substrings *xelent* and *pixel* and *enter* in the last two days.

Note that searches are case insensitive. XELENT is treated the same as xelent or XeLeNt.

You may specify up to 16 terms in the command.

Return Macro Information about Domain

Endpoint: domain/lookup/ Domain Operations Toolset

Returns information about the domain that is specified by either its internal ID or name in the DOMAIN input parameter. If the domain does not exist in our database then the call returns a 404 error. Example:

```
        include("int_ctb.php");
        $dm=new CTB();
        $status=$dm->restCommand("domain/lookup/", array(
                  "key"=> "YOUR_CTB_KEY",
                  "domain"=>"cybertoolbelt.com"
        ));
        if ($status) {
               print $dm->errorMsg()."\n";
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
        $status=$dm->restCommand("domain/lookup", array(
                  "key"=> "YOUR_CTB_KEY",
                  "domain"=>"201294913"
        ));
        if ($status) {
               print $dm->errorMsg()."\n";
               exit;
        }
        $results=$dm->GetResults();
        $ra=json_decode($results,1);
```

This call returns information about the domain as a JSON encoded string as follows (from the above example):

```
Array
(
    [id] => 201294913
    [domain] => cybertoolbelt.com
    [first_seen] => 1386760501
    [last_updated] => 1454614320
    [ipv4] => 6
    [ipv6] => 0
    [subdomains] => 21
    [dns] => 21
    [in_cache] => 1
)
```

Both input formats of the "domain" parameter return the same data.

The fields returned are:

Field	Meaning

id	Internal ID of the domain
domain	Name of the domain
fiirst_seen	Unix-style timestamp of time CTB first saw the domain
last_updated	Unix-style timestamp of time CTB last updated domain information
ipv4	Number of IPv4 addresses related to domain
ipv6	Number of IPv6 addresses related to domain
subdomains	Number of subdomains related to the domain
dns	Number of non-MX non-NS,,non-A, non-AAAA DNS records related to domain
in_cache	0 if DNS has not been updated in last 24 hours, otherwise 1

Return Detail Information about Domain

Endpoint: domain/info/

Returns detailed information about and/or related to a domain. The parameters you pass are:

Field	Meaning	
name	Either the domain name or the ID of the domain	
what	What type of data to return (pick one only):	
	base	General information about the domain
	ns	Name servers associated with the domains
	ip	Related IPv4 and IPv6 information
	whois	Whois record information
	reg	Related registrar information
	dns	Related DNS information
	host	Related subdomain information
	issues	Any issues related to the domain (requires issues toolset)
get_id	When present in a “what=ns” call will return the ID of any each related IP appended to the actual IP address separated by a colon. Ie: “23.44.1.45:34567”	
raw	When present in a “what=whois” call will return the raw (unparsed) Whois record in a subarray named “raw”. Note that not all domains have raw records.	
nodiffs	When present in a “what=whois” call causes only the most recent parsed record to be returned and none of the “diffs” (changes) that occurred in the Whois record.	
nodig	When present in a “what=dns” call will prevent any internet lookup of DNS records during the call.	
max_returns	When present, defines the maximum number of returns of various items. This typically defaults to 250 – 2000 depending on the data type being returned.	

Example:

```
        include("int_ctb.php");
        $dm=new CTB();
        $status=$dm->restCommand("domain/info/",array(
               "key"=> "YOUR_CTB_KEY",
               "name"=>"cybertoolbelt.com",
               "what"="base"
        ));
        if ($status) {
               print $dm->errorMsg()."\n";
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

The results array contains the following fields:

Field	Meaning
name	Domain name
id	Internal ID of the domain
active_status	CTB's best guess as to whether the domain is active. It is almost always correct.
first	The first time CTB saw this domain. Not necessarily when it was created
last	The last time information about the domain was updated in the CTB database
badhost	If non-zero, it represents the number of subdomains in the bad host subdomain dataset. Note that this field is depreciated
emails	An array of emails related to the domain

The following shows output from the above example:

```
(
    [domain_info] => Array
        (
            [name] => cybertoolbelt.com
            [id] => 201294913
            [active_status] => 1
            [first] => 1386760501
            [last] => 1497357735
            [badhost] => 0
            [emails] => Array
                (
                    [0] => Array
                        (
                            [0] => 1645422
                            [1] => who
                            [2] => 1463402809
                            [3] => private_registration@namesbeyond.com
                        )

                    [1] => Array
                        (
                            [0] => 1645422
                            [1] => whoc
                            [2] => 1412097245
                            [3] => private_registration@namesbeyond.com
                        )

                    [2] => Array
                        (
                            [0] => 59671429
                            [1] => soa
                            [2] => 1409667303
                            [3] => hostmaster@cybertoolbelt.com
                        )

                    [3] => Array
                        (
                            [0] => 59671429
                            [1] => who
                            [2] => 1409231787
                            [3] => hostmaster@cybertoolbelt.com
                        )

                )

            [active_ips] => Array
                (
                    [0] => 198.154.119.28
                )

            [active_ns] => Array
                (
                    [0] => ns1.peer1.net
                    [1] => ns2.peer1.net
                )

        )

)
```

Due to historical reasons the emails sub-array does not include named fields. The fields for each array entry (in order) are: email_id, the source of the email address, the UNIX timestamp when CTB made the relationship and finally, the actual email name.

See the “domain/email/” call for the meanings of the email’s source.

Note that the full format of the Whois record returned when “what=whois” is documented in Appendix A.

Get all Detail Information about a Domain

Endpoint: domain/all_info/ Domain Operations Toolset

Returns all information about and/or related to a domain. It is equivalent to all the individual “domain/info/” calls. The parameters you pass are:

Field	Meaning
name	Either the domain name or the ID of the domain
get_id	When present in return the ID of any each related IP appended to the actual IP address separated by a colon. Ie: “23.44.1.45:34567” for NS entries.
raw	When present will return the raw (unparsed) Whois record in a subarray named “raw”. Note that not all domains have raw records.
nodiffs	When present causes only the most recent parsed record to be returned and none of the “diffs” (changes) that occurred in the Whois record.
nodig	When present will prevent any internet lookup of DNS records during the call.
noreg	When present will not return a “reg[]” sub-array as the information is already in the Whois sub-array.
noissues	When present will not return any domain issues information if you are subscribed to the issues data set.
max_returns	When present, defines the maximum number of returns of various items. This typically defaults to 250 – 2000 depending on the data type.

Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“domain/all_info/”, array(
       “key”=> “YOUR_CTB_KEY”,
       “name”=>”cybertoolbelt.com”,
       “noreg”=”y”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

Note that the returned data is an array of all the individual subarrays of the various “what=” types of “domain/info/” calls available. Note also that this call consumes 8 daily quota queries.

This call, under certain extreme circumstances, can take up to a minute to process. It depends on the amount of data being returned and the various relationships of the data.

Domains Related by Email Address

Endpoint: domain/emails/ Domain Operations Toolset

Returns information about domains that are related to the passed email address. The parameters you pass are as follows:

Field	Meaning
email	The email address to use
max_returns	When present, defines the maximum number of returns of various items. This defaults to 500. The maximum value is 10000.

Example:

```
        include("int_ctb.php");
        $dm=new CTB();
        $status=$dm->restCommand(
        domain/emails/",array(
               "key"=> "YOUR_CTB_KEY",
               "email"=>mlewis@xelent.net");
        if ($status) {
               print $dm->errorMsg()."\n";
               exit;
        }
        $results=$dm->getResults();
```

The results array contains the following fields:

Field	Meaning
username	User name part of the passed email address
email_domain	The domain of the passed email address
total	Total number of related domains
domains	A subarray of arrays. Each element describes a single domain as follows:

id	Internal ID of the domain
domain	Domain name
sources	A subarray of the various sources from which the relationship was derived. “whoc” is a whois contact record; “who” is a whois main record; “gen” is general intel; “soa” is an DNS SOA record

The following is sample output from the above example:

```
(
    [username] => mlewis
    [email_domain] => xelent.net
    [total] => 27
    [domains] => Array
        (
            [0] => Array
                (
                    [id] => 11171913
                    [domain] => xelent.net
                    [sources] => Array
                        (
                            [0] => whoc
                            [1] => soa
                            [2] => gen
                            [3] => who
                        )

                )

            [1] => Array
                (
                    [id] => 11172006
                    [domain] => teariffic.net
                    [sources] => Array
                        (
                            [0] => whoc
                            [1] => who
                        )

                )

            [2] => Array
                (
                    [id] => 11172107
                    [domain] => nepadodgeball.org
                    [sources] => Array
                        (
                            [0] => whoc
                            [1] => who
                        )

                )
       }
}
```

Return Live IP Address for Domain

Endpoint: domain/ip/ Domain Operations Toolset

This call performs a live internet IP lookup for the domain whose name or internal domains ID is specified in the DOMAIN input parameter. It will return the first IP it finds even if there are multiple IPs active for the domain. Example:

```
include("int_ctb.php");
$dm=new CTB();
$status=$dm->restCommand("domain/ip/",array(
       "key"=> "YOUR_CTB_KEY",
       "domain=">"cybertoolbelt.com"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
```

The result is a simple text string which represents the IP address as in:

```
198.154.119.28
```

If the string is empty, then the current IP address could not be determined. This call will try to return an IPv4 address first and then an IPv6 address if available.

Use the "domain/all_ips/" call to retrieve all the currently active IP addresses for a domain.

Return All Live IP Addresses for Domain

Endpoint: domain/all_ips/ Domain Operations Toolset

This call performs a live internet IP lookup for the domain whose name or internal domain id is specified in the DOMAIN input parameter. It will return all the currently active IP addresses it finds for the domain. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(domain/all_ips/domain/ip/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “domain=”>”arin.net”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
```

The result is JSON string which represents the IP address(es) found which resolves to:

```
(
    [0] => 199.43.0.44
    [1] => 199.43.0.43
    [2] => 2001:500:4:c000::43
    [3] => 2001:500:4:c000::44
)
```

Note that the call can return both IPv4 and IPv6 addresses.

Return Domain Name by ID

Endpoint: domain/by_id/ Domain Operations Toolset

This call returns the name of the domain whose internal id is passed in the ID input parameter. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“domain/by_id/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “domain”=> 201294913
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->GetResults();
$ra=json_decode($results,1);
```

This call returns the domain name as a plain string as follows:

```
cybertoolbelt.com
```

Return Domain ID using Domain Name

Endpoint: domain/get_id/ Domain Operations Toolset

This call returns the id of the domain whose name is contained in the passed “domain” parameter. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“domain/get_id/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “domain”=> “cybertoolbelt.com”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

This call returns the id as a plain text string as follows:

```
201294913
```

The call will return a status of 404 if the name does not exist in our domains database.

Return Domain Names by ID List

Endpoint: domain/get_names/ Whois Toolset

This call returns the names of the domains whose internal IDs are passed as a comma separated list in the IDS input parameter. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“domain/get_names/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “ids”=> “201294913,456789,11112233”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->GetResults();
$ra=json_decode($results,1);
```

This call returns the domain names as a comma-separated plain string as follows:

```
cybertoolbelt.com,xyz.com,ssss.com
```

SUBDOMAIN COMMANDS

These commands are all related to the subdomains. They begin with the "/subdomain/" endpoint.

Get Subdomains Related to Domain

Endpoint: subdomain/domain/ Domain Operations Toolset

Returns subdomain information related to the domain/FQDN passed in the "domain parameter". Example:

```
include("int_ctb.php");
$dm=new CTB();
$status=$dm->restCommand("subdomain/known/",array(
       "key"=> "YOUR_CTB_KEY",
       "domain"=>"cybertoolbelt.com"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The information is returned as a JSON string which decodes into an array as shown in this example:

```
(
   [host] => Array
       (
           [0] => Array
               (
                   [0] => dropbox
                   [1] => 1390414090
                   [2] => 1400182557
                   [3] => ds
                   [4] => 1391260815
                   [5] => 318196283
                   [6] => Array
                       (
                           [0] => Array
                               (
                                   [0] => 1394038594
                                   [1] => 1061437
                                   [2] => 64.34.181.59
                               )

                       )

               )

           [1] => Array
               (
                   [0] => mail
                   [1] => 1395502964
                   [2] => 1465326670
                   [3] => ct
                   [4] => 1395502964
                   [5] => 497092988
                   [6] => Array
                       (
```

The call returns an array of zero or more related subdomain records. The records are broken down into the following fields (by array index, not by name):

Index	Meaning
0	Subdomain name.
1	When we first saw it (Unix-style time stamp)
2	When we "last" saw it. This is really when we last looked for it and saw it (Unix-style time stamp)
3	What was the source of the subdomain record (depreciated)
4	When CTB related it to the domain (Unix-style time stamp)
5	ID of the subdomain record
6	Array of related IP address records: [0] – When we related the IP address to the subdomain [1] – Internal IP ID numner [2] – The IP address

The input parameters to the call are the following:

Field	Meaning
domain	The domain to process. Either an ID or a name
single_host	If specified, then the DOMAIN field is actually a FQDN and only information about that subdomain is returned.
max_returns	Maximum number of results to return

If SINGLE_HOST is specified then the output looks like the following (all the fields are the same as in the multiple subdomain array). An example of this is a FQDN passed to the call is "api.cybertoolbelt.com":

```
        include("int_ctb.php");
        $dm=new CTB();
        $status=$dm->restCommand("subdomain/known/",array(
               "key"=> "YOUR_CTB_KEY",
               "single_host" => "Y",
               "domain"=>"api.cybertoolbelt.com"
        ));
        if ($status) {
               print $dm->errorMsg()."\n";
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

```
(
    [host] => Array
        (
            [0] => Array
                (
                    [0] => api
                    [1] => 1409843589
                    [2] => 1465326670
                    [3] => ds
                    [4] => 1409890147
                    [5] => 501461582
                    [6] => Array
                        (
                            [0] => Array
                                (
                                    [0] => 1412615309
                                    [1] => 76660647
                                    [2] => 24.229.85.133
                                )

                        )

                )

        )

)
```

Get IP Addresses Related to Subdomain

Endpoint: subdomain/ip/ Domain Operations Toolset

This command is used to return the IP addresses related to a subdomain. Example:

```
include("int_ctb.php");
$dm=new CTB();
$batchid=time();
$status=$dm->restCommand("subdomain/ip/",array(
       "key"=> "YOUR_CTB_KEY",
       "domain"=>"cybertoolbelt.com,
       "subdomains"=>"api,rest"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
```

Example output:

```
(
    [domain] => cybertoolbelt.com
    [subdomains] => Array
        (
            [0] => Array
                (
                    [subdomain] => api
                    [ip_recs] => Array
                        (
                            [0] => Array
                                (
                                    [when_related] => 1412615309
                                    [ip_id] => 76660647
                                    [ip] => 24.229.85.133
                                )
                        )
                    [live] => Y
                )
            [1] => Array
                (
                    [subdomain] => rest
                    [ip_recs] => Array
                        (
                            [0] => Array
                                (
                                    [when_related] => 1412615309
                                    [ip_id] => 902699
                                    [ip] => 64.34.164.179
                                )
                        )
                    [live] => Y
                )
        )
}
```

The subarray *subdomains* contains the individual subdomain records which consists of the subdomain name and zero or more related IP records and a “live” status indicator.

Within each subdomain record is a list of zero or more IP records related to the subdomain that contain the listed fields for each IP address.

The input parameters passed in the call are defined as:

Field	Meaning
domain	The domain name or ID to which the subdomains are related.
subdomains	One or more comma-separated subdomains to process.
max_returns	When present in calls with many possible returned records limits the number actually returned to max_returns. Defaults 250.
noping	When present will prevent an internet IP lookup call. The status of the returned *live* field will be empty.

Search for Subdomains

Endpoint: subdomain/search/ Domain Operations Toolset

Searches for subdomains using the possibly wildcarded “terms” input parameter. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“subdomain/search/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “terms”=>”acuvue%”,
       “max_returns”=>10
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The information is returned as a JSON string which decodes as shown in this example:

```
(
   [0] => Array
       (
           [host] => acuvue
           [domain_id] => 200822129
           [first_seen] => 1326826936
           [last_seen] => 1457370705
           [source] => df
           [when_related] => 1326826936
           [domain] => socialmapps.com
       )

   [1] => Array
       (
           [host] => acuvue
           [domain_id] => 153474354
           [first_seen] => 1363921883
           [last_seen] => 1457370705
           [source] => gi
           [when_related] => 1363921883
           [domain] => skla.pl
       }
```

The call returns an array of zero or more related subdomain records. The records are broken down into the following fields:

Field	Meaning
host	Subdomain name.
domain_id	The ID of the domain that the subdomain is related to
first_seen	When we first saw it (Unix-style time stamp)

last_seen	When CTB “last” saw it. This is really when CTB last looked for it and saw it (Unix-style time stamp)
source	What was the source of the subdomain record (depreciated)
when_related	When CTB related it to the domain (Unix-style time stamp)
domain	The name of the domain it is related to

Note that as many as twice the number of returns as max_returns may be returned. That is because there are two subdomain tables. The “normal” table and the “bad” table. The bad table is comprised of subdomains belonging to domains that have an excessive number of subdomains. Note that the “bad” table is depreciated and will be merged in a future release.

The input parameters to the call are the following:

Field	Meaning
terms	The search string. If it ends in a “%” or “*” character, then the call returns subdomains starting with the string before the wildcard character. Note that this is *not* a substring call ala the “domain/match/” call.
max_returns	Maximum number of results to return. Defaults to 1000.

WHOIS COMMANDS

These commands are all related to the domain Whois subsystem. They begin with the "/whois/" endpoint.

Get Raw Whois Records by ID

Endpoint: whois/raw/ Whois Toolset

Returns all related raw (unparsed) Whois records related to the parsed whois record whose internal ID is specified by ID. Example:

The information is returned as a JSON string as shown in this example:

```
include("int_ctb.php");
$dm=new CTB();
$status=$dm->restCommand("whois/raw/",array(
       "key"=> "YOUR_CTB_KEY",
       "id"=>42923606"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
print_r($ra);

(
    [0] => Array
        (
            [ts] => 2014-09-04 19:57:08
            [whois_server] => whois.networksolutions.com
            [whoisserver_ip] => 205.178.188.12
            [raw] => Domain Name: XELENT.NET
Registry Domain ID:
Registrar WHOIS Server: whois.networksolutions.com
Registrar URL: http://networksolutions.com
Updated Date: 2011-02-20T00:00:00Z

       ...........
       )
)
```

The call returns an array of zero or more records. Each array element contains the following fields:

Field	Meaning
ts	Time stamp when the record was fetched.
whois_server	Name of the Whois server that provided the information.
whoisserver_ip	The IP address of “whois_server”
raw	The actual text record that returned by the Whois server

Get Whois Contacts for a Domain

Endpoint: whois/all_contacts/ Whois Toolset

Returns all Whois contact records that were ever related to the domain whose name passed or internal ID is passed to the call in the “domain” input parameter. Example

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“whois/all_contacts/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “domain”=> ”xelent.net”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The information is returned as a JSON string. See Appendix A for details of the returned records (the “contacts” array).

Domain Whois Search

Endpoint: whois/datamine/ Domain Whois Datamining Toolset

This command is used to perform a datamining operation on the domain Whois database. The following tables lists the possible input parameters:

Field	Meaning
terms	The actual search string/terms. They are defined starting on the next page.
max_returns	Maximum number of returned hits. Defaults to 500.
page_size	Number of records in a paginated page.
token	If page_size is defined this is the unique access token to be used to retrieve results. It is suggested that you use the MD5 values of a salt (ie: “CTB”) concatenated with system time in nanosecond or millisecond resolution.

Search for all Whois records containing the word “mlewis”:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“whois/datamine/”,array(
        “key”=> “YOUR_CTB_KEY”,
        "terms"=>"mlewis"
));
if ($status) {
        print $dm->errorMsg().”\n”;
        exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The information is returned as a JSON string which decodes as follows:

```
(
    [total_hits] => 210
    [returned_hits] => 210
    [keywords] => [mlewis]
    [hits] => Array
        (
            [0] => Array
                (
                    [whois_id] => 63559355
                    [domain_id] => 73276736
                    [score] => 3.1099524
                )

            [1] => Array
                (
                    [whois_id] => 191183262
                    [domain_id] => 127696247
                    [score] => 3.1030643
                )

            [2] => Array
                (
                    [whois_id] => 194880337
                    [domain_id] => 315336954
                    [score] => 3.078696
                )
        )
)
```

The “total_hits” field is the number of whois records that matched the query. The “returned_hits” field is the number of hits returned in the “hits” array. The “keywords” fields is an array of strings that were matched in the query. You can use this field to highlight the results in the display of the result records.

Each element (if any) in the hits array describes a single hit as follows:

Field	Meaning
whois_id	The internal ID of the Whois record that matched the passed terms.
domain_id	The internal ID of the domain
score	The matching relevance score of the match. The results are sorted by this field (highest to lowest).

Use the “domain/by_id/” call to retrieve the actual Whois records.

The example search above was the simplest available. The search function supports a number of logical operations as well as phrase matching, specific field searches, date operations, etc. For example, your terms could be any of the following:

```
terms=mlewis and pixel
terms=mlewis and pixel not “network solutions”
terms=joeblow@gmail.com since last Friday
terms=email contains mlewis
terms=email contains mlewis and (registrar=netowl or registrar=godaddy)
```

As you can see parentheses are supported.

The full list of operators are:

Field	Meaning
AND	The two terms must both be present in the record for the condition to be true. Example: joe AND blow. Both Joe and Blow must exist in a record for it to be selected. Note that the capitalization of the operators is strictly to improve readability for the example. They are case-insensitive in your actual queries.
OR	Either of the terms must be true. Note that in the absence of any specific operator between terms that OR is the default. Example: mike bill sam is identical to mike OR bill OR sam.
NOT	The following term CAN not appear in the record for a record to be selected. Example: joe AND blow NOT godaddy.
CONTAINS	This is a string operator that requires a term and an operand. For example: registrar contains godaddy. This is a full-text operator.
IS (or "=")	This is a string operator that requires an exact match to be true. Example: "state=nj".
Date Operators	See the following table for details.

There are three date fields associated with Whois records: Created, Updated and Expires. You can use the following date operations on these fields:

Field	Meaning
ON (or "=")	Date must match exactly. Examples: updated on 5/1/2015, created=1/1/2015
ON OR AFTER (or ">=" or "SINCE")	This is a greater than or equal to operator.Examples: created on or after last Monday, expires>=march 5,2014, since yesterday. (note that this is equivalent to: created >=3/5/2014 OR updated >=3/5/2014 if today is 3/6/2014).
ON OR BEFORE (or "<=")	This is a less than or equal to operator. Examples: created before last month, expires<=2/8/2012
AFTER (or ">")	Example: expires after 1/1/2020.
BEFORE (or "<")	Example: created after 3/1/2015 AND created before 3/3/2015.

You can specify dates in almost any manner such as: today, last Friday, last year 5/1/2015, March 8, 2011, 2015-03-09, etc.
Note that the capitalization of the operators is strictly to improve readability for the example. They are case-insensitive in your actual queries.

The following are the individual fields that you can search on (can be used as terms with an operator and an operand):

Field	Field Type	Meaning
CREATED	Date	Represents the date when the Whois record was created. Valid for use with all date operators.
UPDATED	Date	Represents the date when the Whois record was last updated (at the time the record was last fetched by us). Valid for use with all date operators.
EXPIRES	Date	Represents the date when the Whois record is due to expire. Valid for use with all date operators.
SERVER	String	Represents the Whois server that was queried for the data. The only valid operator is “is” or its corresponding sugar “=”.
EMAIL	String	Represents any email address associated with the record. This includes any email address found in a contact record for the Whois rcord. The only valid operator is “is” or its corresponding sugar “=”.
STATUS	String_FT	Represents the Whois record’s status. This is a full-text searchable field that supports the “contains” and phrase match syntax.
REGISTRAR	String_FT	Represents the Whois registrar name. This is a full-text searchable field that supports the “contains” and phrase match syntax.
NAME	String_FT	Represents any of the registrant’s names. These names can appear in the main record (as the registrant contact) or any of the other contacts. This is a full-text searchable field that supports the “contains” and phrase match syntax.
ORGANIZATIO N	String_FT	Represents any of the registrant’s organization names. These organization names can appear in the main record (as the registrant contact) or any of the other contacts. This is a full-text searchable field that supports the “contains” and phrase match syntax.
ADDRESS	String_FT	Represents any of the registrant’s street addresses. These addresses can appear in the main record (as the registrant contact) or any of the other contacts. This is a full-text searchable field that supports the “contains” and phrase match syntax.
CITY	String_FT	Represents any of the registrant’s city names. These city names can appear in the main record (as the registrant contact) or any of the other contacts. This is a full-text searchable field that supports the “contains” and phrase match syntax.
STATE	String	Represents any of the registrant’s state names. These state names can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is “is” or its corresponding sugar “=”.
COUNTRY	String	Represents any of the registrant’s country names. These country names

		can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is “is” or its corresponding sugar “=”.
TELEPHONE	String	represents any of the registrant’s telephone number fields. These telephone numbers can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is “is” or its corresponding sugar “=”.
FAX	String	Represents any of the registrant’s fax number fields. These fax numbers can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is “is” or its corresponding sugar “=”.
POSTAL_CODE	String	Represents any of the registrant’s postal code (zip code in the US) fields. These postal codes can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is “is” or its corresponding sugar “=”.

In the above field descriptions there are references to full-text fields (“String_FT”) and non-full-text fields (“String”). The only difference is what operators can be used to search them. For example, the email field is a non-full text field. That means you can only search for the example email address as in:

email=jblow@gmail.com

If the field was a full-text field you could search three different ways:

1. email contains jblow
2. email contains ”jblow@gmail.com”
3. email contains gmail.com

These are only restrictions when you are using individual field operations as part of your query. The entire Whois record and all of its change records (“diffs”) are stored in one large full-text field. This means that any data is searchable using full-text operators. When this becomes a problem is where you submit a search such as the following:

jblow@cyberertoolbelt.com

Expecting to get back records that match jblow@cybertoolbelt.com exactly. What you will, in fact, get back are all the records that match “jblow” or “cybertoolbelt.com. This is because the query processor breaks on the “@” character so your query becomes instead “jblow OR cybertoolbelt OR com”. To do this kind of full-text search use a phrase term (basically put double quotes around the string you want to search on).

PAGINATED SEARCHES

The whois data mining normally returns all the results in a single call. You have the option of doing a paginated search using the *page_size* and *token* parameters. *page_size* specifies the number of results that will be returned at a time. *token* is a unique key used to access those results after the search. The results from a paginated call initially only returns the meta data associated with the search results. You would then use the "whois/get_page/" call to retrieve each page of results.

Example: Search for all Whois records matching "mlewis and xelent.net":

```
        include("int_ctb.php");
        $dm=new CTB();
        $token=md5("ctb".time());
               $status=$dm->restCommand("whois/datamine/", array(
               "key"=> "YOUR_CTB_KEY",
               "terms"=>"mlewis and \"xelent.net\"",
               "max_returns"=>10,
               "page_size"=>4,
               "token"=>$token
        ));
        if ($status) {
               print $dm->errorMsg()."\n";
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

Note that the domain name "xelent.net" is passed to the call with escaped double-quote marks (the escaping is for PHP). That is so that "xelent.net" won't be split into "xelent OR net".

The metadata is returned as a JSON string which decodes as follows:

```
(
    [total_hits] => 12
    [returned_hits] => 10
    [keywords] => [mlewis]
    [pages] => 3
)
```

The metadata is the same as returned with the standard "whois/datamine call/" but with an additional field: *pages.* This field is the number of pages of results that are available for retrieval.

Get Whois Record by ID

Endpoint: whois/by_id/ Whois Toolset

This call returns the whois record whose internal ID is passed in the “whois_id” in the input parameter. Example:

```
include(“int_ctb.php”);
$dm=new ctb();
$status=$dm->restCommand(“whois/by_id/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “whois_id”=> 12345,
       “raw” => “Y”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The information is returned as a JSON string in the same manner as a standard “whois/” call.

The input parameters are:

Field	Meaning
whois_id	The internal CTB whois ID (typically returned via a whois/datamine call)
nodiffs	If set, will prevent the list of record changes from being returned. Only the most recent record will be returned.
raw	When set to “Y” will return the actual Whois text record that was returned by the Whois server as well as the standard parsed Whois record.

```
include(“int_ctb.php”);
$dm=new ctb();
$status=$dm->restCommand(“whois/by_id/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “whois_id”=> 12345,
       “raw” => “Y”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

Get Whois Record by Domain Name

Endpoint: whois/by_name/ Whois Toolset

This call returns the whois record whose internal ID is passed in the "whois_id" in the input parameter. Example:

```
;
include("int_ctb.php");
$dm=new ctb();
$status=$dm->restCommand("whois/by_id/",array(
       "key"=>"YOUR_CTB_KEY",
       "whois_id"=>12345,
       "raw"=>"Y"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

```
include("int_ctb.php");
$dm=new ctb();
$status=$dm->restCommand("whois/by_name/",array(
       "key"=>"YOUR_CTB_KEY",
       "domain"=>"cybertoolbelt.com",
       "raw" => "Y"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
$results=$dm->getResults();
$ra=json_decode($results,1)
```

The information is returned as a JSON string in the same manner as a standard "whois/" call.

The input parameters are:

Field	Meaning
domain	The name of the domain for which to fetch the Whois information for.
nodiffs	If set, will prevent the list of record changes from being returned. Only the most recent record will be returned.
raw	When set to "Y" will return the actual Whois text record that was returned by the Whois server as well as the standard parsed Whois record.

Get Paginated Domain Whois Results Page

Endpoint: whois/get_page/ Domain Whois Datamining Toolset

This command is used to perform a retrieval of a previous “whois/datamine/” paginated operation. The two parameters can be passed to the command:

Field	Meaning
page_number	Page number of results to return. Page numbers start at 1 not 0
token	The token used to create the results set

Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“whois/get_page/”,array(
                “key”=> “YOUR_CTB_KEY”,
                "token"=>$token,
                “page_number”=>3
        ));
        if ($status) {
                print $dm->errorMsg().”\n”;
                exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

The information is returned as a JSON string which decodes as follows:

```
[0] => Array
    (
        [whois_id] => 63559355
        [domain_id] => 73276736
        [score] => 3.1099524
    )

[1] => Array
    (
        [whois_id] => 191183262
        [domain_id] => 127696247
        [score] => 3.1030643
    )
```

The hit results are the same as the non-paginated returned hits.

IP COMMANDS

All the commands in this section are related to IP address operations. They begin with the “/ip/” endpoint.

IP Detailed Information

Endpoint: ip/info/ IP Operations Toolset

This call returns all known information about an IP address that we have in our database. It counts as 6 queries towards your daily quota due to the number of individual calls it makes:

The following details the input parameters:

Field	Meaning
ip	IP address to retrieve information for
nordns	Specifying this parameter will cause the reverse DNS call to be skipped (can take 0.5 – 1.5 seconds to complete)
max_returns	Specifies the maximum number of returns for certain data fields. As a rule don’t set this as the system defaults have been tuned.
nodiffs	When specified prevents record changes from being returned as part of the results.
no_asn	Do not return ASN information
ip_blocks	If ASN information is being returned then specifying this parameter will result in active and inactive IP block information being returned as well as name change history. Note this can result in megabytes of information being returned.

The IP address can be specified as either a dotted quad or the internal ID of the IP address. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“ip/info/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “ip”=>:”64.34.164.179”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The following is sample output from this example:

```
Array
(
    [ip] => 64.34.164.179
    [rdns] => Array
               {
                      [0] => ithreat.xelent.net
               }
    [proxy] => Array
        (
        )

    [tor] => Array
        (
        )

    [vpn] => Array
        (
        )

    [geo_location] => Array
        (
            [0] => Array
                (
                    [0] => Array
                        (
                            [ip_from] => 1076011008
                            [ip_to] => 1076022271
                            [country_code] => US
                            [continent_code] => NA
                            [continent_name] => NORTH AMERICA
                            [time_zone] => EST
                            [region_code] => VA
                            [region_name] => VIRGINIA
                            [owner] => PEER 1 NETWORK INC.
                            [city_name] => HERNDON
                            [county_name] => FAIRFAX
                            [postal_code] => 20171
                            [metro_code] => 511
                            [area_code] => 703
                            [lat] => 38.9696
                            [lon] => -77.3866
                            [source] => 1
                            [when_updated] => 1396652879
                            [updates] => Array
                                (
                                )

                        )

                    [1] => Array
                        (
                            [ip_from] => 1076012032
                            [ip_to] => 1076012236
                            [country_code] => US
                            [country_name] => United States
                            [time_zone] => -04:00
                            [region_code] =>
                            [region_name] => Virginia
                            [owner] => ServerBeach
                            [city_name] => Herndon
                            [lat] => 38.92451
                            [lon] => -77.40187
                            [postal_code] => 20171
                            [metro_code] =>
                            [area_code] => 571/703
                            [domain_name] => serverbeach.com
                            [netspeed] => T1
```

```
                              [idd_code] => 1
                              [mc_code] =>
                              [mn_code] =>
                              [mobile_brand] =>
                               [usage_type] => DCH
                              [source] => 1
                              [when_updated] => 1460034490
                              [updates] => Array
                                  (
                                  )

                          )

                  )

          )

   [domains]=>206171956:a1qualitypharma.com,206171955:boonehvac.com,131111452:ceostrategysuccess.com,
50895486:cirosbistro.com,325556237:cyberintel.report,201294913:cybertoolbelt.com,
166125880:findmyclass.net,239505079:findmyclass4.me,124389781:ftengine.com,11172290:hss4.me,
161537852:hss4.us,308621229:lashable.la,9328979:mickisticki.com,206171957:msucybervettingstudy.net,
11172167:myipaddr.com,11172377:myithreat.com,6255509:nepadodgeball.com,11172107:nepadodgeball.org
164398839:reversedomains.net,11172050:rimrockcottages.com,11172006:teariffic.net,11171913:xelent.net
   [domain_count] => 22
    [subdomains] => Array
        (
            [0] => Array
                (
                    [domain_id] => 201294913
                    [ip_related] => 1412615309
                    [host_id] => 502152275
                    [domain] => cybertoolbelt.com
                    [host] => rest
                    [source] => ds
                    [when_related] => 1410235741
                )
            [1] => Array
                (
                    [domain_id] => 201294913
                    [ip_related] => 1412615309
                    [host_id] => 501799917
                    [domain] => cybertoolbelt.com
                    [host] => tnews
                    [source] => ct
                    [when_related] => 1409933454
                )

            [2] => Array
                (
                    [domain_id] => 11172050
                    [ip_related] => 1445094296
                    [host_id] => 497092991
                    [domain] => rimrockcottages.com
                    [host] => smtp
                    [source] => ct
                    [when_related] => 1395503161
                )


            [20] => Array
                (
                    [domain_id] => 11171913
                    [ip_related] => 1387384192
                    [host_id] => 52861015
                    [domain] => xelent.net
                    [host] => ithreat
                    [source] => gi
                    [when_related] => 1387384192
                )

        )
    [subdomain_count] => 21
    [whois] => Array
        (
            [source] => ARIN

            [range] => 64.34.160.0 - 64.34.175.255
            [size] => 4096
            [desc] =>
            [org_contact] => Array
                (
                    [source] => ARIN
                    [created] => 2007-01-19
                    [changed] => 2007-04-03
                    [handle] => SERVE-33
                    [other_data] => Array
                        (
                            [orgid] => SERVE-33
                            [orgname] => ServerBeach
                            [canallocate] =>
                            [street] => Suite 225-2350 Corporate Park Drive
                            [city] => Herndon
                            [state/prov] => VA
                            [country] => US
                            [postalcode] => 20171
                            [regdate] => 2007-01-19
                            [updated] => 2007-04-03
                            [orgadminhandle] => ZZ4092-ARIN
                            [orgtechhandle] => ZZ4092-ARIN
                            [orgabusehandle] => SNAE-ARIN
                            [source] => ARIN
                        )

                )
            [admin_contact] => Array
                (
                    [handle] =>
                )

            [tech_contact] => Array
                (
                    [source] => ARIN
                    [created] => 2003-09-16
                    [changed] => 2003-09-16
                    [handle] => HOSTM325-ARIN
                    [other_data] => Array
                        (
                            [pochandle] => HOSTM325-ARIN
                            [isrole] => Y
                            [lastname] => Hostmaster
                            [firstname] =>
                            [street] => Array
                                (
                                    [0] => 5150 Broadway
                                    [1] => Suite #620
                                )

                            [city] => San Antonio
                            [state/prov] => TX
                            [country] => US
                            [postalcode] => 78209
                            [regdate] => 2003-09-16
                            [updated] => 2003-09-16
                            [officephone] => +1-210-225-4725
                            [mailbox] => hostmaster@serverbeach.com
                            [source] => ARIN
                        )

                )
            [owner] =>
             [owner_contact] => Array
                 (
                     [handle] =>
                 )
              [created] => 2007-01-19
             [changed] => 2007-01-19
             [city] =>
     }
     [asn_info] => Array
         (
              [badness] => Array
                      {
                              [id] => 8663946,
                              [asn] => 13768,
                              [ip_from] => 86773760,
                              [ip_to] => 3653660671,
                              [date] => "2017-06-16",
                              [abusive] => 70,
                              [running_abusive] => 444,
                              [total] => 1530121,
                              [abuse_score] => 0.004574802,
                              [cumul_abuse_score] => 0.029017312,
                              [abuse_index] => 2.9833143,
                              [abuse_rank] => 6605
                      }
         }
)
```

Notice the last sub-array. It contains information about the current ASN number for the IP as well as any badness information about the ASN. The worst ASNs are ranked 1-100.

ASN Abuse is a measure of the amount of abuse that occurs in a particular ASN (which is a group of IP numbers). It allows you to "judge" the neighborhood where the IP address is located. The numbers may not make much sense upon first glance so here's a handy little cheat sheet:

- **date**. This is date that is the most recent abuse record for the ASN.
- **abuse_rank**. This is where the ASN ranks out of all the ASNs that we process. 1 is the worst.
- **total**. This is the total number of IP addresses contained in the ASN, excluding unusable IP addresses (ie, broadcast addresses).
- **abusive**. This is number of IP addresses that are currently marked as hosting abusive/malicious activity in the ASN.
- **running_abusive**. This is the cumulative one-year running total of abusive IPs seen in the ASN. The running total actually only trails back to April 9, 2017. It is a count of the unique IPs that CTB has abuse information for.
- **abuse_score**. This is a percentage of the number of IP addresses that are abusive in the ASN.
- **cumul_abuse_score**. This is the same percentage of the number of IP addresses that are abusive in the ASN but over the trailing year.
- **abuse_index**. This is a 0 - 10 ranking of where the ASN fits. 0 is no abuse at all and 10 is very bad.
- **ip_from**. This is starting IP address in network format of the ASN.
- **ip_to**. This is ending IP address in network format of the ASN.

Geolocate IP

Endpoint: ip/geolocate/ IP Operations Toolset

This call returns geolocation information about an IP address. The address to geolocate is defined by the "ip" input parameter. Example:

```
include("int_ctb.php");
$dm=new CTB();
$status=$dm->restCommand("ip/geolocate/",array(
       "key"=> "YOUR_CTB_KEY",
       "ip"=> "64.34.164.179"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The resulting JSON decodes to an array as follows:

```
(
    [geo_location] => Array
        (
            [0] => Array
                (
                    [0] => Array
                        (
                            [ip_from] => 1076012032
                            [ip_to] => 1076012236
                            [country_code] => US
                            [country_name] => United States
                            [time_zone] => -04:00
                            [region_code] =>
                            [region_name] => Virginia
                            [owner] => ServerBeach
                            [city_name] => Herndon
                            [lat] => 38.92451
                            [lon] => -77.40187
                            [postal_code] => 20171
                            [metro_code] =>
                            [area_code] => 571/703
                            [domain_name] => serverbeach.com
                            [netspeed] => T1
                            [idd_code] => 1
                            [mc_code] =>
                            [mn_code] =>
                            [mobile_brand] =>
                            [usage_type] => DCH
                            [source] => 1
                            [when_updated] => 1460034490
                            [changes] => Array
                                (
                                )
                        )

                )

        )

)
```

Note that multiple results are possible from different sources. The source field specifies the source. The changes array is a standard field name, value, timestamp listing for each change.

Get AS Number Badness

Endpoint: ip/asn_badness/ IP Operations Toolset

This call returns information about the abuse data for the ASN that the “ip” input parameter is part of. Or the “ip” parameter can be the actual ASN. Using the ASN speeds up the call since a translation does not need to be made. Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“ip/asn_badness/”,array(
               “key”=> “YOUR_CTB_KEY”,
               “ip”=> “17488”
        ));
        if ($status) {
               print $dm->errorMsg().”\n”;
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

The resulting JSON decodes to an array as follows:

```
(
    [badness] => Array
        (
            [id] => 8747846
            [asn] => 17488
            [ip_from] => 453246976
            [ip_to] => 3524443391
            [date] => 2017-06-17
            [abusive] => 503
            [running_abusive] => 3324
            [total] => 919350
            [abuse_score] => 0.054712567
            [cumul_abuse_score] => 0.3615598
            [abuse_index] => 4.530186
            [abuse_rank] => 3389
        )
)
```

ASN abuse is a measure of the amount of abuse that occurs in a particular ASN (which is a group of IP numbers). It allows you to "judge" the neighborhood where the IP address is located. The numbers may not make much sense upon first glance so here's a handy little cheat sheet:

- **date**. This is date that is the most recent abuse record for the ASN.
- **abuse_rank**. This is where the ASN ranks out of all the ASNs that we process. 1 is the worst.
- **total**. This is the total number of IP addresses contained in the ASN, excluding unusable IP addresses (ie, broadcast addresses).
- **abusive**. This is number of IP addresses that are currently marked as hosting abusive/malicious activity in the ASN.

- **running_abusive**. This is the cumulative one-year running total of abusive IPs seen in the ASN. The running total actually only trails back to April 9, 2017. It is a count of the unique IPs that CTB has abuse information for.
- **abuse_score**. This is a percentage of the number of IP addresses that are abusive in the ASN.
- **cumul_abuse_score**. This is the same percentage of the number of IP addresses that are abusive in the ASN but over the trailing year.
- **abuse_index**. This is a 0 - 10 ranking of where the ASN fits. 0 is no abuse at all and 10 is very bad.
- **ip_from**. This is starting IP address in network format of the ASN.
- **ip_to**. This is ending IP address in network format of the ASN.

Get IP Address AS Number

Endpoint: ip/asn_number/ IP Operations Toolset

This call returns the AS number that an IP address belongs to. The IP address is defined in the call as the “ip” parameter. Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“ip/asn_number”,array(
               “key”=> “YOUR_CTB_KEY”,
               “ip”=> “64.34.164.179”
        ));
        if ($status) {
               print $dm->errorMsg().”\n”;
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

The resulting JSON string decodes as an array as follows:

```
(
    [asn] => 13768
)
```

Get AS Numbers for IP Address

Endpoint: ip/asns/ IP Operations Toolset

This call returns AS number records that an IP address belongs to. Example

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“ip/asns/”,array(
        “key”=> “YOUR_CTB_KEY”,
        "ip"=>"168.229.3.2",
        "from_date"=>"2014-01-01"
));
if ($status) {
        print $dm->errorMsg().”\n”;
        exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The input parameters are as follows:

Field	Meaning
ip	This is a standard dotted quad notation IPv4 IP address
from_date	This is an optional date which specifies from which to retrieve the records. If it is not specified then all related records are returned.

The fields returned for each ASN from the call are:

Field	Meaning
id	Internal ID of the record
asn	The ASN number
ip_from	Starting IP number of the block the IP belongs to a 32 bit integer
ip_to	Ending IP number of the block the IP belongs to as a 32 bit integer
size	The number of IP addresses in the block
active	“0” means no, “1” means yes.
ts_first	Date of the first time we saw that ASN record
ts_last	Date of the last time we saw that ASN record

Note that results are returned in reverse date order. That is, newest first – the record that should have the active value set equal to “1”.

The result JSON become an array of subarrays as follows:

```
(
    [0] => Array
        (
            [id] => 1186324
            [asn] => 394488
            [ip_from] => 2833580032
            [ip_to] => 2833612799
            [size] => 32768
            [active] => 1
            [ts_first] => 2017-02-02
            [ts_last] => 2017-03-16
        )

    [1] => Array
        (
            [id] => 778310
            [asn] => 23456
            [ip_from] => 2833580032
            [ip_to] => 2833612799
            [size] => 32768
            [active] => 0
            [ts_first] => 2015-10-24
            [ts_last] => 2017-02-01
        )

    [2] => Array
        (
            [id] => 422014
            [asn] => 1239
            [ip_from] => 2833580032
            [ip_to] => 2833645567
            [size] => 65536
            [active] => 0
            [ts_first] => 2014-12-10
            [ts_last] => 2015-10-23
        )

    [3] => Array
        (
            [id] => 11823
            [asn] => 1239
            [ip_from] => 2833580032
            [ip_to] => 2833643519
            [size] => 63488
            [active] => 0
            [ts_first] => 2014-12-09
            [ts_last] => 2014-12-09
        )

)
```

Get IP Address from ID

Endpoint: ip/by_id/

This call returns the IPv4 number referenced by its internal ID number. Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“ip/by_id/”, array(
               “key”=> “YOUR_CTB_KEY”,
               “id”=> 902699
        ));
        if ($status) {
               print $dm->errorMsg().”\n”;
               exit;
        }
        $results=$dm->getResults();
        print “$results\n”;
```

The result is a simple string as follows:

```
64.34.164.179
```

Check for IP Use Type

Endpoint: ip/type/ IP Operations Toolset

This call returns Information about how an IP might be used (VPN, Proxy, Tor node). Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“ip/type/”,array(
               “key”=> “YOUR_CTB_KEY”,
               “ip”=> “87.159.167.247”
        ));
        if ($status) {
               print $dm->errorMsg().”\n”;
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

There are three primary subarrays returned: “proxy”, “vpn”, and “tor”. Returned information for Vpn:

```
(
    [proxy] => Array
        (
        )

    [tor] => Array
        (
        )

    [vpn] => Array
        (
            [0] => Array
                (
                    [country] => NORWAY
                    [first_seen] => 1456686756
                    [last_seen] => 1474110291
                    [owner] => PureVPN
                )
        )
)
```

The fields returned for each VPN record are:

Field	Meaning
country	The probable country the VPN server is located in
first_seen	The time (as a Unix-style timestamp) that CTB first registered the VPN
last_seen	The last time CTB updated the record as a Unix-style timestamp
owner	Ownership information for the VPN

The resulting JSON for the “tor” subarray decodes as follows:

```
(
    [proxy] => Array
        (
        )

    [tor] => Array
        (
            [0] => Array
                (
                    [name] => mYour
                    [router_port] => 9001
                    [directory_port] => 9030
                    [roles] => Array
                        (
                            [0] => Fast
                            [1] => Guard
                            [2] => HS Dir
                            [3] => Running
                            [4] => Stable
                            [5] => V2 Dir
                            [6] => Valid
                        )
                    [uptime] => 933361
                    [version] => Tor 0.2.4.27
                    [contact] => Random Person &lt;myTor@t-online.de&gt;
                    [last_seen] => 1445361830
                    [active] => inactive
                    [diffs] => Array
                        (
                            [0] => Array
                                (
                                    [field] => contact_info
                                    [old_value] => Random Person &lt;myTor@t-online.de&gt;
                                    [when_changed] => 1444998274
                                    [new_value] =>
                                )
                            [1] => Array
                                (
                                    [field] => contact_info
                                    [old_value] =>
                                    [when_changed] => 1445001865
                                    [new_value] => Random Person &lt;myTor@t-online.de&gt;
                                )
                        )
                )
        )
    [vpn] => Array
        (
        )
)
```

The fields returned for each Tor record are:

Field	Meaning
name	Self-reported name
router_port	Metadata
directory_port	Metadata
roles	Metadata
uptime	Uptime in seconds

version	Tor version
contact	Contact information
last_seen	The last time CTB updated the record
active	“Y” if node is active
diffs	Subarray of change records to the information about the node

This is the information returned for “proxy” records:

```
(
    [proxy] => Array
        (
            [0] => Array
                (
                    [country] => ID
                    [port] => 8080
                    [first_seen] => 1448658363
                    [last_seen] => 1493184064
                    [type] =>
                    [protocol] => http
                    [anonymity] => Low
                    [score] => 1
                )

            [1] => Array
                (
                    [country] =>
                    [port] => 80
                    [first_seen] => 1480141412
                    [last_seen] => 1480314189
                    [type] =>
                    [protocol] => http
                    [anonymity] =>
                    [score] => 1
                )

        )

    [tor] => Array
        (
        )

    [vpn] => Array
        (
        )

)
```

The fields returned for each Proxy record are:

Field	Meaning
country	The probable country the proxy server is located in as its 2-character code
port	Port to contact the proxy server on
first_seen	The time (as a Unix-style timestamp) that CTB first registered the proxy server

last_seen	The last time CTB updated the record as a Unix-style timestamp
type	Type of proxy
protocol	Type of protocol accepted
anonymity	Level of anonymity afforded by the proxy
score	Currently unused

The data retured for these three types of IP addresses can vary in the information contaned from record to record. They contain as much information as we can glean. We do not claim to known about all proxies, Vpns and Tor nodes. But we generally know a lot.

Check IP Against Known Tor Exit Nodes

Endpoint: ip/tor/ IP Operations Toolset

This call returns the status of an IP address as a Tor exit node. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“ip/is_tor/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “ip”=> “87.159.167.247”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
print $results;
```

This call returns a “yes” or “no” response a plain string.

Check if IP is a Known Proxy

Endpoint: ip/proxy/ IP Operations Toolset

This call returns the status of an IP address as a proxy. Example:

```
include("int_ctb.php");
$dm=new CTB();
$status=$dm->restCommand("ip/is_proxy/",array(
       "key"=> "YOUR_CTB_KEY",
       "ip"=> "87.159.167.247"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
$results=$dm->getResults();
print $results;
```

This call returns a "yes" or "no" response a plain string.

Translate IP ID to IP Address (IPv4 or IPv6)

Endpoint: ip/translate/ IP Operations Toolset

This call is a more general version of the “ip/by_id/” call. It works with either an IPv4 or IPv6 internal ID number. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“ip/translate/,array(
       “key”=> “YOUR_CTB_KEY”,
       “id”=> 455553)
       “protocol”=> 4
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
print “$results\n”;
```

The result is a simple string which represents either the IPv4 or IPv6 human readable address as shown here:

```
64.34.164.179
```

You pass in a protocol type of “4” if the passed ID is an IPv4 address or a type of “6” for an IPv6 ID.

Get Domains Related to IP Address

Endpoint: ip/domains/ IP Operations Toolset

This call returns any domains related to the passed IP address. The input parameters are as follows:

Field	Meaning
ip	This is a standard dotted quad notation IPv4 IP address
max_returns	This is an optional parmeter which specifies the maximum number of domains to return. It defaults to 500. You can set it as high as 10000.
Keywords	An optional list of comma-separated keywords that if defined, must exist in the domain names to be returned. Ie: If cyber,intel is specified in this parameter then only domains which contain either “cyber” or “intel” will be returned as results.

Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“ip/domains/”,array(
               “key”=> “YOUR_CTB_KEY”,
               “ip”=> “64.34.164.179”
               “max_returns”=> 5
        ));
        if ($status) {
               print $dm->errorMsg().”\n”;
               exit;
        }
        $results=$dm-getResults();
        $ra=json_decode($results,1);
```

The resulting JSON is an array with two fields:

```
(
    [domains] =>
325556237:cyberintel.report:1443931764,323641417:rimrockcottages.com:
1437044652,308621229:lashable.la:1429416556,201294913:cybertoolbelt.com:1409890147,
239505079:findmyclass4.me:1391260815
    [domain_count] => 23
)
```

The first field is the list of domain entries. Each entry has two parts: The domain name and its internal ID (separated by a colon). The second field is the total number of domains that are related to the IP address.

Regardless of the value of max_returns, “domain_count” in the results will always contain the number of related domains in the database.

Process ICG Sensor Blocklist

Endpoint: ip/blocklist/ Update Operations Toolset

This call is used to process a blocklist from a CTB sensor into IP issue records. The blocklist is a list of one or more IP addresses that have been doing something they shouldn't. The typical blocklist might be SSH credential attacks. Example:

```
include("int_ctb.php");
$dm=new CTB();
$status=$dm->restCommand("ip/blocklist/",array(
       "key"=> "YOUR_CTB_KEY",
       "blocklist"=>"64.34.164.179\n64.34.17.210",
       "type"=>"bl",
       "attack_type"=>"ssha",
       "info"=>"CTB sensor alert"
);
$results=$dm->getResults();
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
```

The input parameters are:

Field	Meaning
blocklist	This is a list of or more IP addresses separated by new line characters ("\n").
type	This is the type of list. Currently only "bl" is supported.
attack_type	The type of attack the blocklist represents. See the list in the "ip/save_issue/" write up. The same codes are used.
info	This is either a text field or an encoded JSON string representing any additional optional information about the entries.
report_time	The optional time the incident occurred. If not supplied it defaults to the current time.

Save IP Issue

Endpoint: ip/save_issue/

This call saves or modifies an IP issue record. Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“ip/save_issue/”,array(
       “key”=> “YOUR_CTB_KEY”,
       “id”=> 0,
       “ip”=>“64.34.164.179”,
       “reason”=>”phish”,
       “report_time”=>1490233456,
       “removed”=>0,
       “info”=>”{\”url\”:\”xxx.phish.com\”,\”source\”:\”CTB\”}”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The input parameters are:

Field	Meaning
id	This is a non-zero record number if you are editing an existing record. For a new record this must be zero.
ip	This is the IP address for which to create the IP issue record.
reason	This is a short code that determines the type of issue. The codes are listing in the following table.
report_time	This is a Unix-style timestamp that is the time that the issue was discovered.
removed	If the issue has been resolved or removed, then this is the Unix-style timestamp when that happened.
info	This is a generalized text field containing additional information about the issue. It can be a simple text string or a JSON string of key/value pairs as shown in the above example. Nothing is done with this field except to return it on “ip/info” or “ip/get_issues/” calls. For example, the CTB web site displays this information in its IP display.

The result of the call for creating a new issue is either a 0 (the issue has already been reported for that IP) or the record ID of the new record. In the case of an edit operation (ID!=0) the response is always an empty string unless there is an error.

The following table lists the currently defined reason codes and their meaning. You can create your own codes using these guidelines:

1. Codes should be short and sweet abbreviations. They are limited to 16 characters in size.
2. The general format of the code is WHAT DESCR. For example. *mw* stands for malware. That is a code by itself. *mwds* stands for MalWare Distribution Site.
3. If you create a new code please advise technical support (support@cybertoolbelt.com) so that it can be included in this table and suitable modifications can be made to the consumers of the data (ie, the GUI).

REASON code table:

code	Meaning
bn	Botnet
bncc	Botnet command and control server
cc	Generic command and control
cfh	Cold Fusion hack
drh	Drupal hack
feodocc	FEODO command and control
joh	Joomla hack
mw	Malware related
mwds	Malware distribution site
mwcc	Malware command and control
phish	Phishing related
plugx C&C	PLUGX command and control
pony loader c&c	Pony load command and control
ransomware	Ransomware related
rwcc	Ransomware command and control
rwds	Ransomware distribution site
rwps	Ransomware ??? site
spam	Spam related
sshc	SSH compromise
trojan	Trojan related

Trojan.Fareit	??
vncc	VNC compromise
wph	Wordpress root hack
Sqli	SQL Injection

Get Known Issues for IP Address

IP Detailed Information

Endpoint: ip/issue/ Issues Toolset

This call returns any issues related to an IP address. Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“ip/issue/”,array(
               “key”=> “YOUR_CTB_KEY”,
               “ip”=> “64.34.164.179”
        ));
        if ($status) {
               print $dm->errorMsg().”\n”;
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

The resulting JSON become an array as follows:

```
(
    [0] => Array
        (
            [id] => 3949716
            [ip] => 64.34.164.179
            [reason] => botcc
            [report_time] => 1481308868
            [info] => {"url":"cybertoolbelt.com","source":"CTB development"}
            [removed] => 0
        )

    [1] => Array
        (
            [id] => 3949714
            [ip] => 64.34.164.179
            [reason] => test
            [report_time] => 1481308566
            [info] => This is a test record
            [removed] => 0
        )

)
```

The fields returned are:

Field	Meaning
id	This the internal ID for the record
ip	IP address that the issue relates to

reason	This is a short code that describes the type of issue.
report_time	This is a Unix-style timestamp that is the time that the issue was discovered.
removed	If the issue has been resolved or removed, then this is the Unix-style timestamp when that happened.
info	This is a generalized text field containing additional information about the issue. It can be a simple text string or a JSON string of key/value pairs as shown in the above example. Nothing is done with this field except to return it on “ip/info/” or “ip/get_issues calls/”. For example, the GUI displays this information for IP display.

The REASON codes are listed in the “/ip/save_issue/” write-up.

IP Detailed AS Number Information

Endpoint: ip/asn_info/ IP Operations Toolset

This call returns information about the ASN that an IP address belongs to. There is an optional "ip_blocks" input parameter which specifies that the call should return IP block information. Example:

```
include("int_ctb.php");
$dm=new CTB();
$status=$dm->restCommand("ip/asn_info/",array(
       "key"=> "YOUR_CTB_KEY",
       "ip"=>"64.34.164.179"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The resulting JSON represents an array as follows:

```
(
    [error] => 0
    [ip] => 64.34.164.179
    [ips] => Array
        (
        )

    [asns] => Array
        (
            [0] => Array
                (
                    [source] => ARIN
                    [asn] => 13768
                    [name] => PEER1
                    [description] =>
                    [city] =>
                    [country] =>
                    [created] => 2002-06-10
                    [updated] => 2012-02-24
                    [status] =>
                    [handles] => Array
                        (
                            [Organization] => PER1
                            [TechHandle] => Array
                                (
                                    [0] => ZP55-ARIN
                                )

                        )

                )

        )

    [contacts] => Array
        (
        )

    [source] => ARIN
)
```

IP Whois

Endpoint: ip/whois/ IP Operations Toolset

This call returns Whois information about an IP address. There are several parameters:

Field	Meaning
ip	This is the IP address for which to return the information. It is a required parameter
get_asn	Gets the AS number (ASN) for the passed IP address
ip_blocks	Returns the associated IP blocks information for the ASN. This can return very large (multiple megabyte) results
get_geo	Returns geolocation information about the IP address
get_rdns	Returns any RDNS information for the IP address
get_domains	Returns any associated domains (up to max_returns) and the total count
get_subdomains	Returns any associated subdomains (up to max_returns) and the total count
max_returns	Limits the number of returned domain and subdomains return to this value (defaults to 2000)

Example:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“ip/whois”,array(
       “key”=> “YOUR_CTB_KEY”,
       “ip”=> “64.34.164.179”
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The result is returned as a JSON string whose format is detailed in Appendix B.

IP Whois Search

Endpoint: ip/datamine/ IP Whois Datamining Toolset

This command is used to perform a datamining operation on the IP Whois database. The following are the input parameters can be passed to the command:

Field	Meaning
terms	The actual search string/terms. They are defined starting on the next page
max_returns	Maximum number if returned hits. Defaults to 500

Search for all IP Whois records containing the word “peer1”:

```
include(“int_ctb.php”);
$dm=new CTB();
$status=$dm->restCommand(“ip/datamine/”, array(
       “key”=> “YOUR_CTB_KEY”,
       "terms"=>"peer1"
));
if ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

The information is returned as a JSON string which decodes as follows:

```
(
    [total_hits] => 3607
    [returned_hits] => 500
    [keywords] => [peer1]
    [hits] => Array
        (
            [0] => Array
                (
                    [whois_id] => 63559355
                    [ip_star] => 73276736
                    [ip_end] => 73276736
                    [score] => 3.1099524
                )

            [1] => Array
                (
                    [whois_id] => 191183262
                    [domain_id] => 127696247
                    [score] => 3.1030643
                )

            [2] => Array
                (
                    [whois_id] => 194880337
                    [domain_id] => 315336954
                    [score] => 3.078696
                )
        )
)
```

The total_hits field is the number of whois records that matched the query. The returned_hits field is the number of hits returned in the "hits" array. The keywords fields is an array of strings that were matched in the query. You can use this field to highlight the results in any displays. Each element (if any) in the hits array describes a single hit as follows:

Field	Meaning
whois_id	The internal ID of the Whois record that matched the passed terms.
start_ip	Starting IP of the block that matches
end_ip	Ending IP of the block that matches.
score	The matching relevance score of the match. The results are sorted by this field (highest to lowest).

The following are the individual fields that you can search on (can be used as terms with an operator AND an operand):

Field	Field Type	Meaning
CREATED	Date	Represents the date when the Whois record was created. Valid for use with all date operators.

UPDATED	Date	Represents the date when the Whois record was last updated (at the time the record was last fetched by us). Valid for use with all date operators.
EXPIRES	Date	Represents the date when the Whois record is due to expire. Valid for use with all date operators.
IP_FROM	IP	Represents the starting IP address in a block to search for.
IP_TO	IP	Represents the ending IP address in a block to search for.
EMAIL	String	Represents any email address associated with the record. This includes any email address found in a contact record for the Whois rcord. The only valid operator is “is” or its corresponding sugar “=”.
STATUS	String_FT	Represents the Whois record’s status. This is a full-text searchable field that supports the “contains” and phrase match syntax.
PERSON	String_FT	Represents the person’s information.
NAME	String_FT	Represents any of the registrant’s names. These names can appear in the main record (as the registrant contact) or any of the other contacts. This is a full-text searchable field that supports the “contains” and phrase match syntax.
ORGANIZATIO N	String_FT	Represents any of the registrant’s organization names. These organization names can appear in the main record (as the registrant contact) or any of the other contacts. This is a full-text searchable field that supports the “contains” and phrase match syntax.
ADDRESS	String_FT	Represents any of the registrant’s street addresses. These addresses can appear in the main record (as the registrant contact) or any of the other contacts. This is a full-text searchable field that supports the “contains” and phrase match syntax.
CITY	String_FT	Represents any of the registrant’s city names. These city names can appear in the main record (as the registrant contact) or any of the other contacts. This is a full-text searchable field that supports the “contains” and phrase match syntax.
STATE	String	Represents any of the registrant’s state names. These state names can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is “is” or its corresponding sugar “=”.
COUNTRY	String	Represents any of the registrant’s country names. These country names can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is “is” or its corresponding sugar “=”.
TELEPHONE	String	represents any of the registrant’s telephone number fields. These telephone numbers can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is “is” or its corresponding sugar “=”.

FAX	String	Represents any of the registrant's fax number fields. These fax numbers can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is "is" or its corresponding sugar "=".
POSTAL_CODE	String	Represents any of the registrant's postal code (zip code in the US) fields. These postal codes can appear in the main record (as the registrant contact) or any of the other contacts. The only valid operator is "is" or its corresponding sugar "=".

In the above field descriptions there are references to full-text fields ("String_FT") and non-full-text fields ("String"). The only difference is what operators can be used to search them. For example, the email field is a non-full text field. That means you can only search for the example email address as in:

email=jblow@gmail.com

If the field was a full-text field you could search three different ways:

1. email contains jblow
2. email contains "jblow@gmail.com"
3. email contains gmail.com

These are only restrictions when you are using individual field operations as part of your query. The entire Whois record and all of its diffs are stored in one large full-text field. This means that any data is searchable using full-text operators. When this becomes a problem is where you submit a search such as the following:

jblow@cyberertoolbelt.com

Expecting to get back records that match jblow@cybertoolbelt.com exactly. What you will, in fact, get back are all the records that match "jblow" or "cybertoolbelt.com. This is because the query processor breaks on the "@" character so your query becomes instead "jblow OR cybertoolbelt OR com".

Get IP Reverse DNS (rDNS)

Endpoint: ip/info/ IP Operations Toolset

This call is used to return all rDNS records that CTB ever knew was associated with a particular IP. Example:

```
        include("int_ctb.php");
        $dm=new CTB();
        $status=$dm->restCommand("/ip/get_rdns/" array(
               "key"=> "YOUR_CTB_KEY",
               "ip"=>"81.245.32.82"
        ));
        $results=$dm->getResults();
        if ($status) {
               print $dm->errorMsg()."\n";
               exit;
        }
        $ra=json_decode($results,1);
        print_r($ra);
```

The input parameter is the IP address to lookup.

The following is sample output:

```
(
    [rdns] => Array
        (
            [0] => Array
                (
                    [id] => 10044
                    [rdns_id] => 10044
                    [ip_id] => 83891738
                    [when_related] => 1493982182
                    [proto] => 4
                    [rdns] => 82.32-245-81.adsl-dyn.isp.belgacom.be
                )

        )

)
```

The input parameters are:

Field	Meaning
id	The internal ID of the rDNS to IP record
rdns_id	Internal ID of the rDNS record
ip_id	The internal ID of the IP record
proto	Either 4 for IPv4 or 6 for IPv6

rdns	The rDNS name for the IP address

Note that this call does a live lookup of the rDNS information when you call it. The results are returned in reverse time order (newest first).

EMAIL COMMANDS

Commands in this section concern data related to email addresses. They all start with the "/email/" endpoint.

Get Domains Related to Email Address

Endpoint: email/info/ Domain Operations Toolset

This call returns information associated with the passed email address. Example:

```
include("int_ctb.php");
$dm=new CTB();
$status=$dm->restCommand("
       email/info/" array(
       "key"=> "YOUR_CTB_KEY",
       "email"=> "mlewis@xelent.net"
));
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
$results=$dm->getResults();
$re=json_decode($results,1);
```

The above example would return the following:

```
(
    [username] => mlewis
    [email_domain] => xelent.net
    [total] => 27
    [domains] => Array
        (
            [0] => Array
                (
                    [id] => 11171913
                    [domain] => xelent.net
                    [sources] => Array
                        (
                            [0] => whoc
                            [1] => soa
                            [2] => gen
                            [3] => who
                        )

                )
            [1] => Array
                (
                    [id] => 11172107
                    [domain] => nepadodgeball.org
                    [sources] => Array
                        (
                            [0] => whoc
                            [1] => who
                        )

                )
```

REPORT COMMANDS

Report commands are used to feed and retrieve Cyber alert information into the CTB database from external sources.

Save Cyber Report

Endpoint: report/cyber/ Update Operations Toolset

This command is used to upload an ICG Cyber Practice report to CTB: Example

```
include("int_ctb.php");
$dm=new CTB();
$ra=array("title"=>"Test",
       "key"=> "YOUR_CTB_KEY",
       "ip"=> "64.34.164.179{,
       "report_date"=>"2016-03-10 15:12:54",
       "severity"=>"5",
       "type"=>"hack",
       "report"=>"The world is on fire!"
));
$status=$dm->restCommand("report/cyber/",$ra);
if ($status) {
       print $dm->errorMsg()."\n";
       exit;
}
$results=$dm->getResults();
$ra=json_decode($results,1);
```

In the passed parameters list the severity varies from 0 – 5 (0 being least, 5 the most severe).

The report_date field is the date and time of the report in "YYYY-MM-DD HH:MM:SS" date format.

Type is the type of the report limited to 16 characters.

Report is the actual report being made. It is limited to 16mb characters in size.

The enterprise_id field is a list of one or more CTB enterprise ID numbers for whom the report pertains. Each element in the list is separated by "|" characters as in "43|12|14". An enterprise_id number of "0" specifies that the report is applicable to all users who have access to these reports.

The result of the call is either a single field "error" with any backend error code or the field "status" which should contain the value "acknowledged".

RELATED COMMANDS

A “related” command returns information about things that are related to another thing that don’t easily fall into another category. They begin with the “/related/” endpoint.

Get Domains Related to Name Server

Endpoint: domains/ns/ Domain Operations Toolset

Returns information about domains related to a particular domain name server specified by “name” input field. Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“related/ns/”,array(
               “key”=> “YOUR_CTB_KEY”,
               “name”=> “ns1.peer1.com”
        ));
        if ($status) {
               print $dm->errorMsg().”\n”;
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

This call will return information about domains related to the domain name server as a JSON formatted string which decodes as follows:

```
Array
(
    [total_count] => 2432
    [records] => eggspecial.com|1359925103|308967, fatonmedia.com|1360010815|1009984,
       breadpackaging.com|1360121203|2751209, appitology.com|1360203360|4057480,
       appotology.com|1360203360|4057488,fsbobooster.com|1360094051|2323536,
       sepguardado.com|1360686842|17901211, glarre.com|1360398476|9693215,
       azhutchinsn.com|1360571532|14896809…………
)
```

The “total_count” field contains the count of the total number of domains related to the name server. The call defaults to returning no more than 500 domains. However, using the “max_returns” input parameter, you can set anywhere from 1 to 25000 as the number to be returned.

The records field is a string where each record is separated by commas. There are three fields in each record separated by vertical bar (“|”) characters. The fields in the record are as follows:

1. domain name.
2. when it was related to the name server by CTB as a Unix time stamp.

3. Internal ID of the domain.

Get Domains Related to MX Server

Endpoint: domains/mx/ Domain Operations Toolset

Returns information about domains related to a particular MX server specified by the “name” input parameter. Example:

```
        include(“int_ctb.php”);
        $dm=new CTB();
        $status=$dm->restCommand(“related/mx/”,array(
               “key”=> “YOUR_CTB_KEY”,
               “name”=> “vmail.dial-up.net”
        ));
        $status=$dm->domainManager(“related/mx/ vmail.dial-up.net”);
        if ($status) {
               print $dm->errorMsg().”\n”;
               exit;
        }
        $results=$dm->getResults();
        $ra=json_decode($results,1);
```

This call will return information about domains related to the MX server as a JSON formatted string which decodes as follows:

```
Array
(
    [total_count] => 2
    [records] => isadsl.co.za|1408590073|228944465,tip-offs.org|1416075405|13754769
)
```

The “total_count” field contains the count of the total number of domains related to the name server. The call defaults to returning no more than 500 domains. However, using the “max_returns” input parameter, you can set anywhere from 1 to 25000.

The records field is a string where each record is separated by commas. There are three fields in each record separated by vertical bar (“|”) characters. The fields in the record are as follows:

1. domain name.
2. when it was related to the MX server by CTB as a Unix time stamp.
3. Internal ID of the domain.

MISCELLANOUS COMMANDS

These commands are typically in the “get/” end point and return various data.

Get New Domains

Endpoint: get/new_domains/ New Domains Toolset

This call returns the new domains since a user-specified time. Note that the time cannot be older that 4000 seconds from the current time. Example:

```
include(“int_ctb.php”);
$dm=new DM();
$status=$dm->restCommand("get/new_domains/",
       "post",array(
              “key”=> “YOUR_CTB_KEY”,
              "from_time”=>1510223634
              )
       );
If ($status) {
       print $dm->errorMsg().”\n”;
       exit;
}
$results=$dm->GetResults();
$ra=json_decode($results,1);
```

The results are returned as a JSON string that equates to the following array structure:

```
(
    [0] => Array
        (
            [domain] => elit01.ir
            [tld] => ir
        )

    [1] => Array
        (
            [domain] => crafts.ec
            [tld] => ec
        )

    [2] => Array
        (
            [domain] => pluglab.co.tz
            [tld] => tz
        )

    [3] => Array
        (
            [domain] => clickwebhosting.in
            [tld] => in
        )

    [4] => Array
        (
            [domain] => furbz.co.uk
            [tld] => uk
        )
```

This example shows the first 5 elements of the array returned. The fields returned in each element are as follows:

Field	Meaning
domain	The domain name of the new domain
tld	The TLD of the new domain

Note that the definition of "new" is new to CTB. This is usually the same as new to the DNS. However, in the case of ccTLDs is may mean that CTB has never seen it before.

APPENDIX A: DOMAIN WHOIS RECORD FORMAT

The following documents the format of the Whois record without the contacts array. That is documented on the next page but it is part of the whois array as a subarra

```
                  [contacts] => Array
                        (
                            [0] => Array
                                (
                                    [registrant] => Array
                                        (
                                            [email] => mlewis@xelent.net
                                            [name] => Xelen.net Ltd
                                            [organization] => Xelen.net Ltd
                                            [street1] => 728 Main Street
                                            [street2] =>
                                            [street3] =>
                                            [street4] =>
                                            [city] => Stroudsburg
                                            [state] => PA
                                            [country] => US
                                            [fax] => +1.9999999999
                                            [sfax_ext] =>
                                            [telephone] => +1.9999999999
                                            [telephone_ext] =>
                                            [postal_code] => 18360
                                            [md5] => 868d61c0df845a711d7760011ee6ba13
                                            [id] => 139026581
                                            [changes] => Array
                                                (
                                                    [0] => Array
                                                        (
                                                            [field] => country
                                                            [old_value] =>
                                                            [when] => 1452690178
                                                            [new_value] => US
                                                        )

                                                )
                                        )

                                )

                            [1] => Array
                                (
                                    [administrative] => Array
                                        (
                                            [email] => mlewis@xelent.net

                     Etc…
```

```
                                    [old_value] => clientTransferProhibited
                                    [when] => 1406326794
                                    [new_value] =>
                                )
```

There are up to five types of contacts. This example shows the Registrant and the Administrative records (or part of them). There is also Technical, Billling and Zone.

```
                            [3] => Array
                                (
                                    [field] => email
                                    [old_value] => mlewis@xelent.net
                                    [when] => 1406326794
                                    [new_value] =>
                                )

                        )

                )
```

Note that the MD5 value is passed for each contact so that you can determine which contacts are identical. It is very common for a single contact to be the Registrant, Administrative and Technical contacts.

APPENDIX B: IP WHOIS RECORD FORMAT

The following documents the format of the IP Whois record without the ASN information.

```
        (
            [error] => 0
            [ip] => 64.34.164.179
            [ips] => Array
                (
                    [0] => Array
                        (
                            [source] => ARIN
                            [cidr] => Array
                                (
                                    [0] => 64.34.160.0/20
                                )

                            [handle] => NET-64-34-160-0-1
                            [name] => PEER1-SERVERBEACH-02A
                            [ip_range] => Array
                                (
                                    [0] => 64.34.160.0
                                    [1] => 64.34.175.255
                                )

                            [description] =>
                            [city] =>
                            [country] =>
                            [parent] => NET-64-34-0-0-1
                            [origin] =>
                            [nameservers] =>
                            [status] =>
                            [type] => reallocation
                            [created] => 2007-01-19
                            [updated] => 2007-01-19
                            [handles] => Array
                                (
                                    [Organization] => SERVE-33
                                    [TechHandle] => Array
                                        (
                                            [0] => HOSTM325-ARIN
                                        )

                                )

                        )

                )

            [asns] => Array
                (
                )
            [contacts] => Array
                (
                    [0] => Array
                        (
                            [source] => ARIN
                            [handle] => SERVE-33
                            [class] => Organization
                            [name] => ServerBeach
                            [description] =>
                            [type] =>
```

```
                                  [street] => Suite 225-2350 Corporate Park Drive
                        [city] => Herndon
                        [state] => VA
                        [postal_code] => 20171
                        [country] => US
                        [email] =>
                        [allocate] =>
                        [referral] =>
                        [created] => 2007-01-19
                        [updated] => 2007-04-03
                        [handles] => Array
                            (
                                [OrgAdminHandle] => Array
                                    (
                                        [0] => ZZ4092-ARIN
                                    )

                                [OrgTechHandle] => Array
                                    (
                                        [0] => ZZ4092-ARIN
                                    )

                                [OrgAbuseHandle] => Array
                                    (
                                        [0] => SNAE-ARIN
                                    )

                            )

                    )

                [1] => Array
                    (
                        [source] => ARIN
                        [handle] => HOSTM325-ARIN
                        [class] => Contact
                        [name] => Hostmaster
                        [description] => Array
                            (
                            )

                        [street] => 5150 Broadway, Suite #620
                        [city] => San Antonio
                        [state] => TX
                        [postal_code] => 78209
                        [country] => US
                        [phone_office] => +1-210-225-4725
                        [phone_mobile] =>
                        [phone_fax] =>
                        [email] => hostmaster@serverbeach.com
                        [role] => Y
                        [created] => 2003-09-16
                        [updated] => 2003-09-16
                        [handles] => Array
                            (
                            )

                    )
            )

        [source] => ARIN
    )
```

APPENDIX C: PHP INTERFACE PROGRAM

This is the source code of the int_ctb.php API interface class. It is offered on an “as-is” basis with no guarantees as to its fitness for a particular purpose or usage:

```
 <?php
//
//	Interface to CyberTOOLBELT REST API - V4
//	Copyright 2017 by cybertoolbelt.com
//	All Rights Reserved.
//
//	Command syntax:
//		$dm=new CTB();
//		$dm->ctbManager(COMMAND);
//
//	Returms:
//		0 on success else error code
class CTB {
	var $name;
	var $server;
	var $port;
	var $results;
	var $error;
	var $lastaction;
	var $errText;
//
//	Constructor
//
function __construct() {
	$dsn="rest-api.cybertoolbelt.com;2476;restMgr";
//	$dsn="127.0.0.1;2476;restMgr";
	list($this->server,$this->port,$this->name)=explode(";",$dsn,3);
}
function setPort($p) {
	$this->port=$p;
}
//
//	Return error code
//
function getError() {
	return $this->error;
}
//
//	Return results
//
function getResults() {
	return $this->results;
}

//
//	Return last error message
//
function errorMsg() {
	if ($this->errText!="") return $this->errText;
	return "Unknown error code: ".$this->error;
}
//
//      Talk to the API
//
function restCommand($cmd,$ia="") {
        $base = "https://".$this->server.":2476/api/v2/";
        $ca=explode("/",$cmd);
        $route="";
        $x=count($ca);
        for ($i=0;$i<$x-1;$i++) {
                $route.=$ca[$i]."/";
        }
        $terms=urlencode($ca[$x-1]);
        $url="$base$route";
        $curl_result = $curl_err = '';
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        if (is_array($ia)) {
                curl_setopt($ch, CURLOPT_POST, 1);
                $ra=array();
                foreach ($ia as $k=>$v) {
                        $ra[$k]=urlencode($v);
                }
                $str=http_build_query($ra);
                curl_setopt($ch, CURLOPT_POSTFIELDS, $str);
        } else {
                curl_setopt($ch, CURLOPT_POST, 0);
                $url.=$terms;
        }
        curl_setopt($ch, CURLOPT_HEADER, 0);
//@@    curl_setopt($ch, CURLOPT_VERBOSE, 1);
//@@    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);
//@@    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, FALSE);
//@@    curl_setopt($ch, CURLOPT_SSLVERSION, 1);
        curl_setopt($ch, CURLOPT_TIMEOUT, 0);
        curl_setopt($ch, CURLOPT_URL, $url);
        $res = curl_exec($ch);
        $curl_err = curl_error($ch);
        $httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
        curl_close($ch);
        $this->results=$res;
        $this->error=0;
        $this->errText="";
        if (substr($httpCode,0,1)=='4' || substr($httpCode,0,1)=="5") {
                $ra=@json_decode($res,1);
                if (isset($ra['error'])) $this->errText=$ra['error'];
                $this->error=$httpCode;
        }
        return $this->error;
}
}
```

APPENDIX D: PYTHON INTERFACE TO CYBERTOOLBELT API

The following code is a Python 2.x interface to CTB's restful API. It is customer supplied. CTB makes no warrantees or representations as to its utility or fitness for any purpose. It is supplied "as-is" and you use it at your own risk.

```
"""

CyberToolBelt API connector

"""

__version__ = 0.1

__author__ = 'CTB Client'

import httplib

import urllib

import ssl

import json

import socket


# Error classes for catching custom errors in the code.

class CTBError(Exception): pass

class IPError(CTBError): pass

class ASNError(CTBError): pass

class CTBConnection(httplib.HTTPSConnection):

      """

      Custom SSLv2 connections to https://rest-api.cybertoolbelt.com

      """

      def __init__(self):

             self.host = 'rest-api.cybertoolbelt.com'

             self.port = 2476

             httplib.HTTPSConnection.__init__(self, self.host, self.port)

      def connect(self):

             self.sock = ssl.wrap_socket(ssl.socket(), ssl_version=ssl.PROTOCOL_SSLv2)

             self.sock.connect((self.host, self.port))
```

```
class CTBResponse(object):
    """
    Decoded response from https://rest-api.cybertoolbelt.com
    """
    def __init__(self, response):
        self.headers = dict(response.getheaders())
        self.status = response.status
        self.reason = response.reason
        self.data = response.read()
        self.original_response = response
        if self.headers.get('content-type') == 'application/json':
            self.json = json.loads(self.data)
        else:
            self.json = None
    def __repr__(self):
        return '<{}.{} Status:{} {} Length: {}>'.format(
            self.__class__.__module__,
            self.__class__.__name__,
            self.status,
            self.reason,
            self.headers.get('content-length')
        )

class CTBApi(object):
    """
    Base API for CyberToolbelt
    """
    def __init__(self, CTBKey):
        self.key = CTBKey
    def __get(self, path, params=None):
        if params is None:
            params = {'key': self.key}

        conn = CTBConnection()

        conn.request('GET', '{}?{}'.format(path, urllib.urlencode(params)))

        return CTBResponse(conn.getresponse())

    def __post(self, path, params):

        conn = CTBConnection()

        conn.request('POST', '{}?key={}'.format(path, self.key),
urllib.urlencode(params))

        return CTBResponse(conn.getresponse())

    def __valid_asn(self, id):

        try:

            int(id)

        except ValueError:

            return False

        else:

            return True

    def __valid_ip(self, ip):

        try:

            socket.inet_aton(ip)

        except socket.error:

            return False

        else:

            return True

    def whois(self, id=None, ip=None, domain=None, refresh=False):

        if id:

            if self.__valid_asn(id):

                return self.__get('/whois/{}'.format(id), {'key':self.key,
'lookup': 'y' if refresh else 'n'})

            else:

                raise ASNError('Invalid ID.')

            elif ip:

                if self.__valid_ip(ip):
```

```
                                    return self.__get('/whois/ip/{}'.format(ip),
{'key':self.key, 'lookup': 'y' if refresh else 'n'})

                    else:

                            raise IPError('Invalid IP Address.')

                    elif domain:

                            return self.__get('/whois/domain/{}'.format(domain),
{'key':self.key, 'lookup': 'y' if refresh else 'n'})

                    else:

                            raise CTBError('Must specify id, ip, or domain.')

        def ip_related_domains(self, ip, refresh=False):

                if self.__valid_ip(ip):

                        return self.__get('/ip/domains/{}'.format(ip), {'key':self.key, 'lookup':
'y' if refresh else 'n'})

                else:

                        raise IPError('Invalid IP Address.')

        def find_subdomains(self, domain, refresh=False):

                return self.__get('/domains/subdomains/{}'.format(domain), {'key':self.key,
'lookup': 'y' if refresh else 'n'})

        Api = CTBApi('<YOUR_API_KEY_HERE>')


"""

import cybertoolbelt

response = cybertoolbelt.Api.whois(domain="google.com")

data = response.json

"""
```