**RESTful API for Keylime (*v2*)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *tenant -> cloud verifier (CV)* | GET (retrieve) | PUT (replace/edit) | POST (new) | DELETE (remove) |
| /v2/instances/ | Get list of instances from CV |  |  |  |
| /v2/instances/*{instance\_id:UUID}* | Get status of instance instance\_id from CV |  | Add new instance instance\_id to CV  Requires JSON body:  ***{***  ***“v” : key,***  ***“ip” : ipaddr,***  ***“port” : int,***  ***“operational\_state” : int,***  ***“public\_key” : key,***  ***“tpm\_policy” : json,***  ***“vtpm\_policy” : json,***  ***“metadata” : json,***  ***“ima\_whitelist” : json,***  ***}*** | Terminate instance instance\_id |
| /v2/instances/*{instance\_id:UUID}*/reactivate |  | Start instance instance\_id  *For an already bootstrapped instance\_id node* |  |  |
| /v2/instances/*{instance\_id:UUID}*/stop |  | Stop cv polling on instance\_id, but don’t delete  *For an already started instance\_id* |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *verifier/tenant -> node* | GET (retrieve) | PUT (replace/edit) | POST (new) | DELETE (remove) |
| /v2/keys/ |  |  |  |  |
| /v2/keys/pubkey | Retrieves node’s public key |  |  |  |
| /v2/keys/vkey |  |  | Send v\_key to node  Requires JSON body:  ***{***  ***“encrypted\_key”: key,***  ***}*** |  |
| /v2/keys/ukey |  |  | Send u\_key to node (with optional payload)  Requires JSON body:  ***{***  ***“auth\_tag” : hmac,***  ***“encrypted \_key”: key,***  ***“payload”: b64,*** (*opt*)  ***}*** |  |
| /v2/keys/verify  ex.)  /v2/keys/verify/challenge/# | Get confirmation of bootstrap key derivation  Required parameters:  ***challenge : int*** |  |  |  |
| /v2/quotes/ |  |  |  |  |
| /v2/quotes/integrity  ex.)  */v2/quotes/integrity/nonce/#/mask/#/vmask/#/partial/#/* | Get integrity quote from node  Required parameters:  ***nonce : int***  ***mask : bitmask***  ***vmask : bitmask***  ***partial : bool*** |  |  |  |
| /v2/quotes/identity  ex.)  */v2/quotes/identity/nonce/#/* | Get identity quote from node  Required parameters:  ***nonce : int*** |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *verifier/node/tenant -> registrar* | GET (retrieve) | PUT (replace/edit) | POST (new) | DELETE (remove) |
| /v2/instances/ | Get list of registered instances |  |  |  |
| /v2/instances/{instance\_id:UUID} | Get AIK of instance instance\_id  *Typically used by verifier/tenant* |  | Add instance instance\_id to registrar  Requires JSON body:  ***{***  ***“ek” : key,***  ***“ekcert” : cert,***  ***“aik” : key,***  ***}*** | Remove instance instance\_id from registrar |
| /v2/instances/{instance\_id:UUID}/activate |  | Activate physical instance instance\_id  Requires JSON body:  ***{***  ***“auth\_tag” : hmac,***  ***}*** |  |  |
| /v2/instances/{instance\_id:UUID}/vactivate |  | Activate virtual instance instance\_id  Requires JSON body:  ***{***  ***“deepquote” : b64,***  ***}*** |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Tenant webapp* | GET (retrieve) | PUT (replace/edit) | POST (new) | DELETE (remove) |
| /v2/nodes/ | Get ordered list of nodes (by state) |  |  |  |
| /v2/nodes/{instance\_id:UUID} | Get details of specified instance\_id | Start instance instance\_id  *For an already bootstrapped instance\_id node* | Add new instance instance\_id to CV  Requires JSON body:  ***{***  ***“ip” : ipaddr,***  ***“keyfile\_data” : base64,***  ***“keyfile\_name” : string,*** (*opt*)  ***“file\_data” : base64,***  ***“file\_name” : string,*** (*opt*)  ***“ca\_dir” : string,***  ***“ca\_dir\_pw” : string,***  ***“include\_dir\_data” : base64,***  ***“include\_dir\_name” : string,***  ***}*** | Terminate instance instance\_id |
| /v2/logs | Get terminal log data |  |  |  |
| /v2/logs/{logType:string}    ex.)  */v2/logs/tenant/pos/#/* | Get terminal log data for given logType  Possible parameters:  ***pos : int,*** (*opt*) |  |  |  |

**RESTful API Responses for Keylime (*v2*)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *tenant -> cloud verifier (CV)* | GET (retrieve) | PUT (replace/edit) | POST (new) | DELETE (remove) |
| /v2/instances/ | “results” = {  “uuids” = [*uuid1*, *uuid2*, …]  } |  |  |  |
| /v2/instances/*{instance\_id:UUID}* | “results” = {  “operational\_state”,  “v”,  “ip”,  “port”,  “tpm\_policy”,  “vtpm\_policy”,  “metadata”,  “ima\_whitelist”,  } | “results” = {} | “results” = {} | “results” = {} |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *verifier/tenant -> node* | GET (retrieve) | PUT (replace/edit) | POST (new) | DELETE (remove) |
| /v2/keys/ |  |  |  |  |
| /v2/keys/pubkey | “results” = { “pubkey” } |  |  |  |
| /v2/keys/vkey |  |  | “results” = {} |  |
| /v2/keys/ukey |  |  | “results” = {} |  |
| /v2/keys/verify | “results” = { “hmac” } |  |  |  |
| /v2/quotes/ |  |  |  |  |
| /v2/quotes/integrity  ex.)  */v2/quotes/integrity/nonce/#/mask/#/vmask/#/partial/#/* | “results” = {  “quote”,  “pubkey”, (*if not partial*)  } |  |  |  |
| /v2/quotes/identity  ex.)  */v2/quotes/identity/nonce/#/* | “results” = {  “quote”,  “pubkey”,  } |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *verifier/node/tenant -> registrar* | GET (retrieve) | PUT (replace/edit) | POST (new) | DELETE (remove) |
| /v2/instances/ | “results” = {  “uuids” = [*uuid1*, *uuid2*, …]  } |  |  |  |
| /v2/instances/{instance\_id:UUID} | “results” = {  “aik”,  “ek”,  “ekcert”,  “regcount”,  “provider\_keys”, (*ONLY IF VIRTUAL*)  } |  | “results” = {  “blob” = *EncEK(AIK)*,  } | “results” = {} |
| /v2/instances/{instance\_id:UUID}/activate |  | “results” = {} |  |  |
| /v2/instances/{instance\_id:UUID}/vactivate |  | “results” = {} |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Tenant webapp* | GET (retrieve) | PUT (replace/edit) | POST (new) | DELETE (remove) |
| /v2/nodes/ | “results” = {  “uuids” = [*uuid1*, *uuid2*, …]  } |  |  |  |
| /v2/nodes/{instance\_id:UUID} | “results” = {  “id”,  “operational\_state”,  “v”,  “ip”,  “port”,  “tpm\_policy”,  “vtpm\_policy”,  “metadata”,  “ima\_whitelist”,  } | “results” = {} | “results” = {} | “results” = {} |
| /v2/logs |  |  |  |  |
| /v2/logs/{logType:string}    ex.)  */v2/logs/tenant/pos/#/* | “results” = {  “log” = [*string*, *string*, …]  *(lines)*  } |  |  |  |

**JSON Response Format (v2)**

|  |
| --- |
| {  “code” : int,  “status” : string,  “results” : JSON,  } |

**HTTP Status Codes**

|  |  |  |
| --- | --- | --- |
| *Code (Number)* | Code (Textual) | Description |
| 200 | Success (OK) | Results return as requested |
| 201 | Created | A resource was created as requested |
| 400 | Bad request | Your request was incorrect / invalid / malformed |
| 403 | Forbidden | You do not have the required permissions |
| 404 | Not found | API request error |
| 405 | Method not allowed | Requested method not allowed for this resource (provides ‘Allow’ header) |
| 409 | Conflict | Request could not be completed due to resource state conflict |
| 500 | Internal error | Internal problem (e.g., server, database) |
| 501 | Not implemented | Request method not yet supported for this resource |