# strongSwan - Bug #909

## HMAC error from key > 64 bytes in libstrongswan/plugins/hmac

27.03.2015 08:52 - Johan Andersson

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | 27.03.2015 |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | Martin Willi | **Estimated time:** | 0.00 hour |
| **Category:** | libstrongswan | | |
| **Target version:** | 5.3.0 | | |
| **Affected version:** | 5.1.3 | **Resolution:** | Fixed |

**Description**

There is a bug in the libstrongswan/plugins/hmac implementation.
When using keys larger than 64 bytes, the HMAC result will be faulty.

The get_mac method ends with starting to hash over the ipad. If calling set_key method with key > 64 bytes after get_mac, the hash to reduce key size is started with the result from the ipad hash.
So, before doing that hash, the hash functions needs to be reset.

Changing:
METHOD(mac_t, set_key, bool, private_mac_t *this, chunk_t key) { int i; u_int8_t buffer[this->b]; memset(buffer, 0, this->b); if (key.len > this->b) { /* if key is too long, it will be hashed */ if (!this->h->get_hash(this->h, key, buffer)) { return FALSE; } } ...
to:
METHOD(mac_t, set_key, bool, private_mac_t *this, chunk_t key) { int i; u_int8_t buffer[this->b]; memset(buffer, 0, this->b); if (key.len > this->b) { /* if key is too long, it will be hashed */ this->h->reset(this->h); if (!this->h->get_hash(this->h, key, buffer)) { return FALSE; } } ...

Solved the problem.

BR
Johan Andersson

### Associated revisions

**Revision bfb029c8 - 27.03.2015 15:53 - Martin Willi**

hmac: Reset the underlying hasher before doing set_key() with longer keys

The user might have done a non-complete append, having some state in the hasher.

Fixes #909.

### History

**#1 - 27.03.2015 15:38 - Martin Willi**

*- Assignee set to Martin Willi*

**#2 - 27.03.2015 16:02 - Martin Willi**

*- Status changed from New to Closed*

*- Target version set to 5.3.0*

*- Resolution set to Fixed*

Hi Johan,

Thanks for your bug report. I've addressed this issue with the referenced commit.

The error actually triggers only if the hasher has some state, i.e. hmac has called in append mode. This is a rather exotic use, hence this has not been detected so far.

I've extended our crypto-tester to include such an API call sequence. Beside the *hmac* plugin, the *xcbc* plugin was also affected, which I fixed as well. There are some other plugins I'll check and fix (*af-alg* seems to be affected, too).

Regards
Martin

#### #3 - 30.03.2015 09:32 - Johan Andersson

Hi

This is actually not an exotic case. Yes, that this is triggered by a hmac get_key after an hmac get_mac is.
But we found this problem due to the fact that we could not get a strongswan stack with this code to negotiate with a strongswan stack using openssl library instead (for keys > 64 bytes).
So the bug is normally triggered (I think) by that the prf+ for getting the keys leaves the hasher in a state, that then the hmac get_key continues from.
I guess that running two identically configured strongswan stack will work, since the hasher state will be the same on both sides (maybe that is why this has not been found...)

BR
Johan

#### #4 - 30.03.2015 10:02 - Martin Willi

I agree, as HMAC always hashes the ipad key to the state, this occurs for any set_key() with more than 64 bytes issued to a PRF previously used.

As we use 32 byte nonces, PRF+ gets exactly 64 bytes between two strongSwan instances, so this is in many cases no problem.

However, with EAP methods that return MSKs larger than 64 bytes, or with PSKs > 64 bytes in both IKEv1 and IKEv2 this can yield wrong key material.

Regards
Martin