Compromising confidential compute and then fixing it

Ben Hania & Yair Netzer

Ben Hania

Senior Security Researcher,
Microsoft

Yair Netzer

Principal Security Research Manager,
Microsoft

And many more

Moti Markovitz
Maxime Villard
Vladimir Abramzon
Andrey Markovytch
Eli Cohen Nehemia
Ada Ostrokol

Moshe Levi
Avishai Redelman
Tomer Shem-Tov
Nagaraju Kodalapura
Truc Nguyen
Hareesh Khattri
And others…

Edge+Platform Security Fundamentals

Spoiler

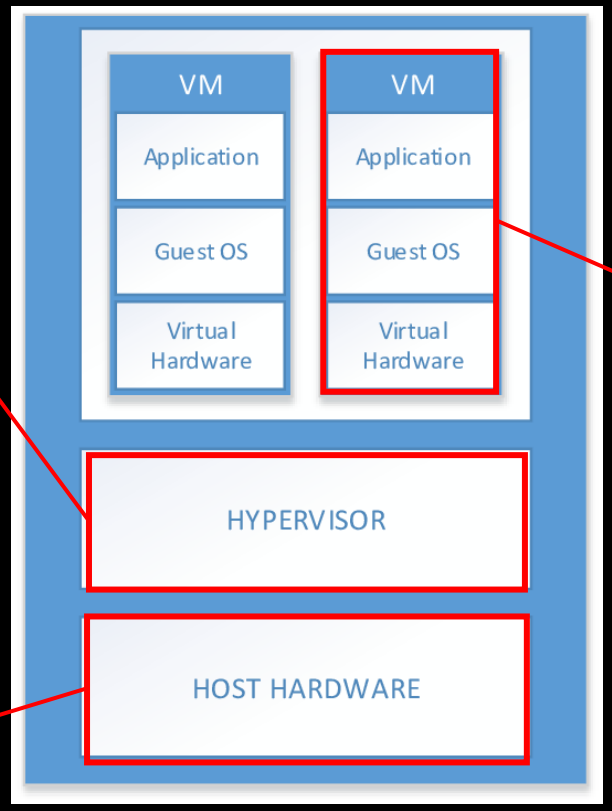
We have CVEs

Slides to CVE

(Very) High Level Cloud Architecture

Microsoft Azure

aws



But what if you don't want to trust the infrastructure provider?

Confidential Compute 101

HW support: AMD SEV-SNP, Intel TDX

Data Security and IP Protection

Protect apps and data from attack, tampering, or theft.

Privacy and Compliance

Strengthen data confidentiality and regulatory compliance.

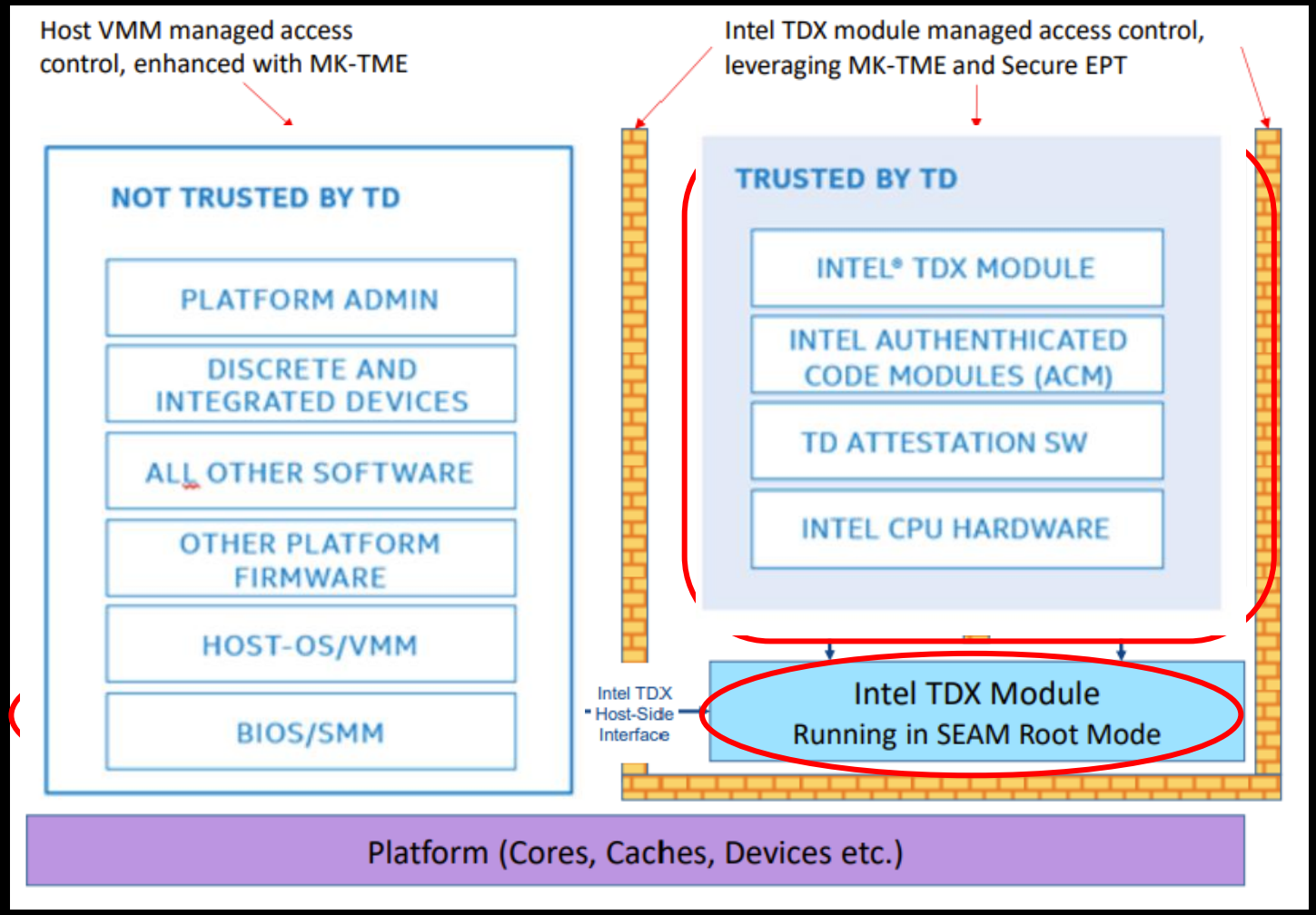
Data Sovereignty and Control

Prohibit access by cloud providers or other tenants. Add safeguards to data sovereignty and governance.

Confidential AI

Safeguard your AI data and models by providing robust isolation, integrity, and confidentiality.

Intel TDX



Glossary:

- VMM – Virtual Machine Manager: Hypervisor
- TDX – Trusted Domain Extension: Broker between VMM and confidential VMs
- SEAM – Secure Arbitration Mode
- TD – Trusted Domain: Confidential VM

Attestation

```
typedef st
{
    report
    /**
     * Add
     * Inc
     */
    tee_tc
    td_inf
} td_repor
```

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD's ATTRIBUTES */
    uint64_t        xfam; /**< TD's XFAM**/
    measurement_t   mr_td; /**< Measurement of the initial contents of the TD */
    /**
     * 48 Software defined ID for additional configuration for the software in the TD
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Software defined ID for TD's owner */
    /**
     * Software defined ID for owner-defined configuration of the guest TD,
     * e.g., specific to the workload rather than the runtime or OS.
     */
    measurement_t   mr_owner_config;
    measurement_t   rtmr[NUM_OF_RTMRS]; /**<  Array of NUM_RTMRS runtime extendable measurement registers */
    measurement_t   servtd_hash;

    uint8_t         reserved[64];
} td_info_t;
```

Generate report

Research Goals

Scope: Intel TDX 1.5 new features

1. Make a malicious TD appear valid (i.e., forge attestation)
2. Access secrets within a customer TD

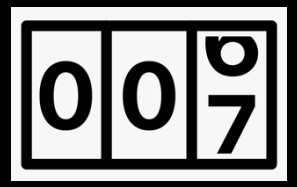
Previous research into Intel TDX 1.0 was done by GPZ:

securing-the-unseen-vulnerability-research-in-confidential-computing.pdf

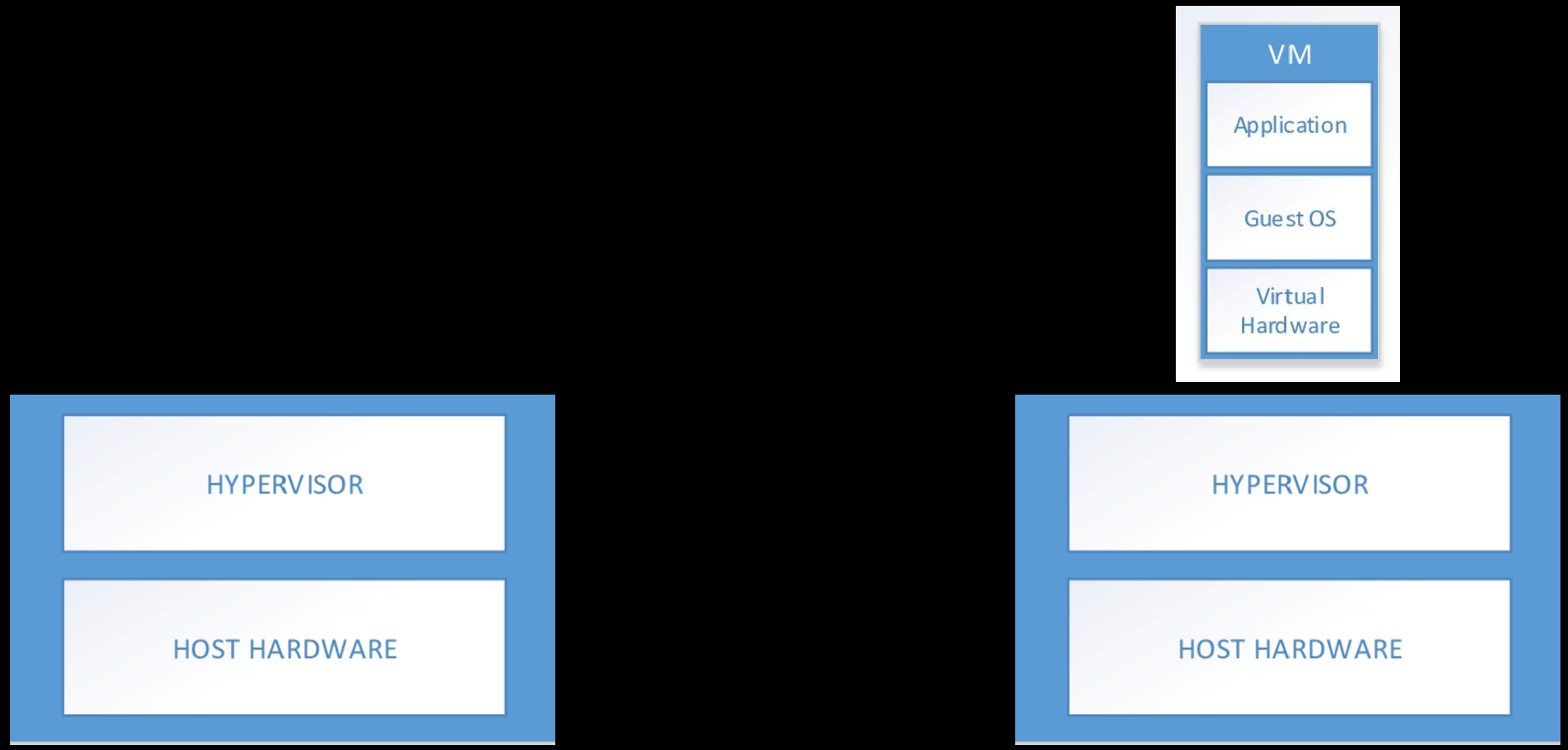
The goal:
Forge Attestation

The method: Live migration

Slides to CVE



Live Migration



Live Migration – in Confidential compute

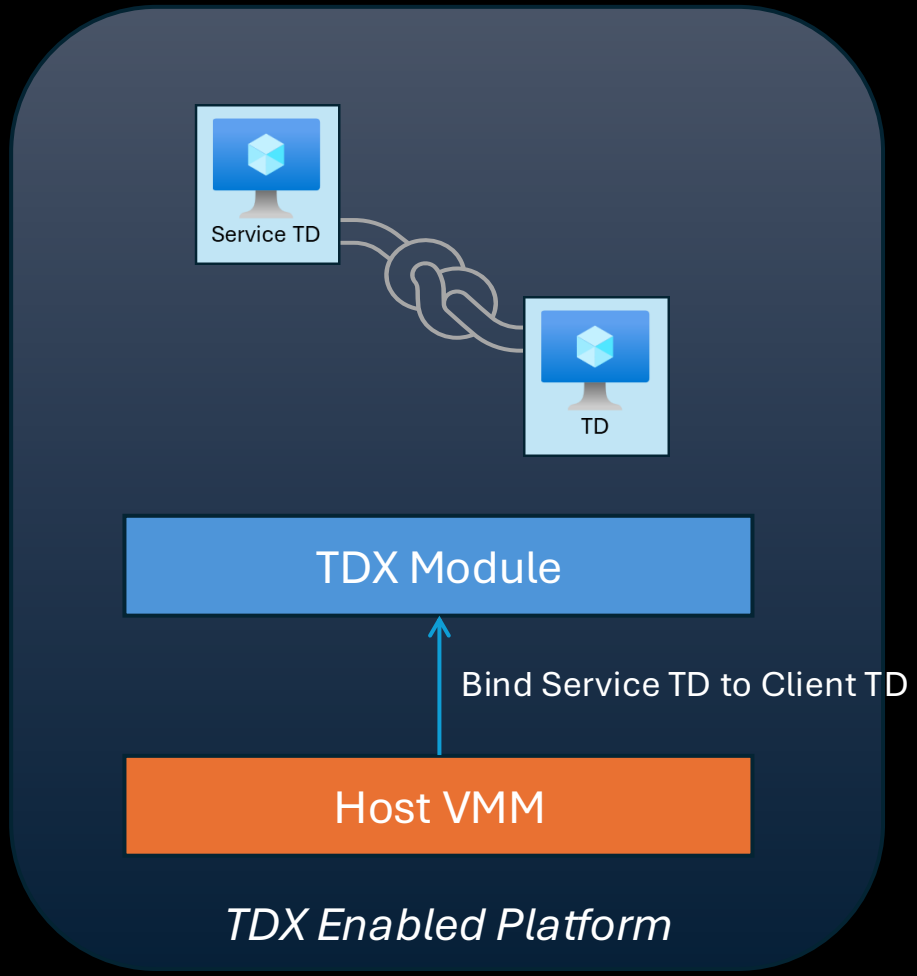
What will we do in TDs?

- No VMM access to TD memory
- TD encryption is tied to a specific TDX instance
- Need some transfer mechanism between TDX devices

Enter: Service TD

- Binding Table, used for under the
- Only one type of service TD – **Migration TD**, or **MigTD** for short

UUID	Service TD unique identifier
INFO HASH	Hash of the Service TD's TD_INFO
…	…



Service TD

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD's ATTRIBU
    uint64_t        xfam; /**< TD's XFAM**/
    measurement_t   mr_td; /**< Measurement of th
    /**
     * 48 Software defined ID for additional con
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Software defin
    /**
     * Software defined ID for owner-defined con
     * e.g., specific to the workload rather tha
     */
    measurement_t   mr_owner_config;
    measurement_t   rtmr[NUM_OF_RTMRS]; /**<  Arr
    measurement_t   servtd_hash;

    uint8_t         reserved[64];
} td_info_t;
```

Binding Table

UUID	Service TD unique identifier
INFO HASH	Hash of the Service TD's TD_INFO
…	…

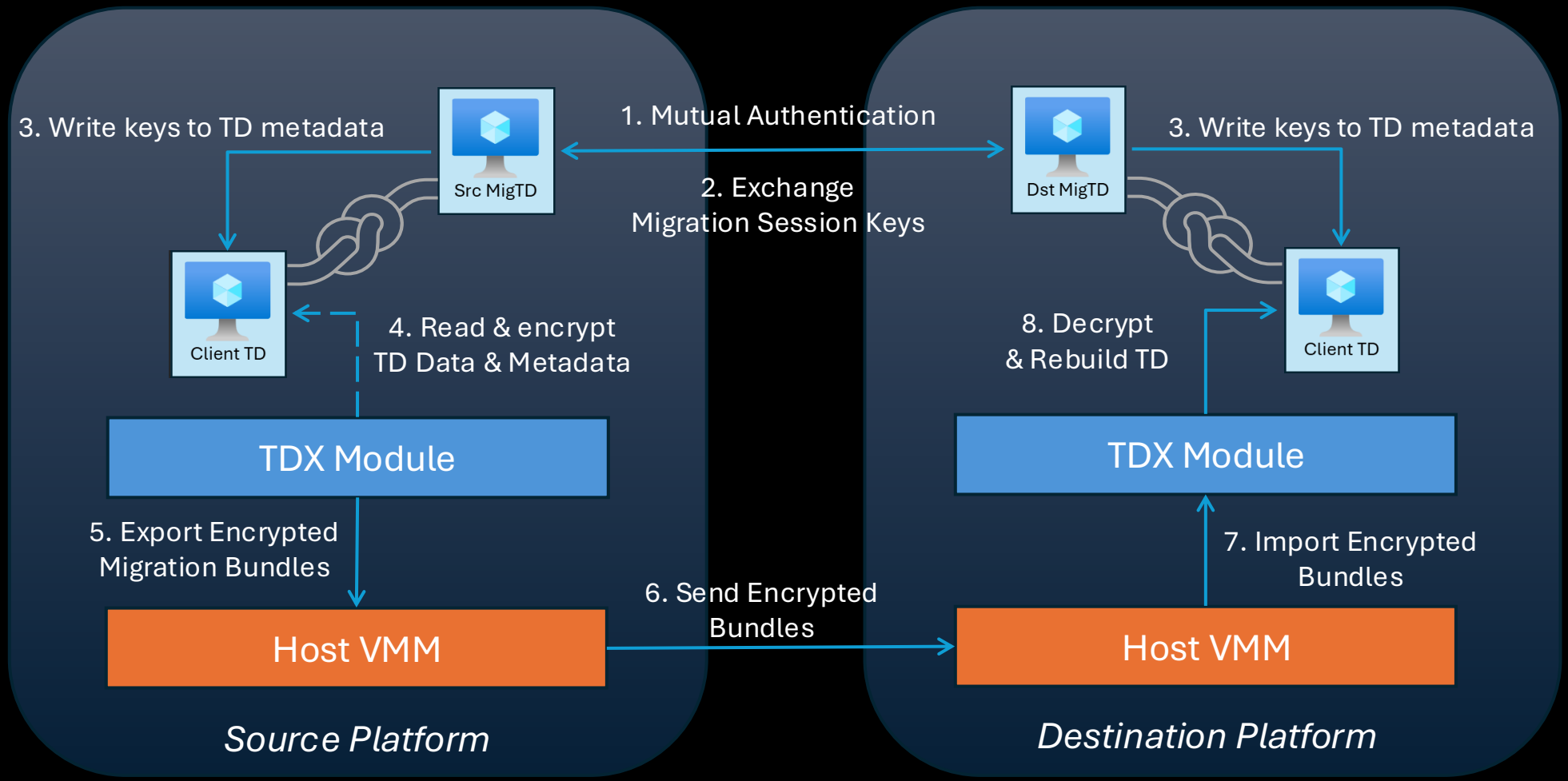
Service TD

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD's ATTRIBU
    uint64_t        xfam; /**< TD's XFAM**/
    measurement_t   mr_td; /**< Measurement of th
    /**
     * 48 Software defined ID for additional con
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Software defin
    /**
     * Software defined ID for owner-defined con
     * e.g., specific to the workload rather tha
     */
    measurement_t   mr_owner_config;
    measurement_t   rtmr[NUM_OF_RTMRS]; /**<  Arr
    measurement_t   servtd_hash;

    uint8_t         reserved[64];
} td_info_t;
```

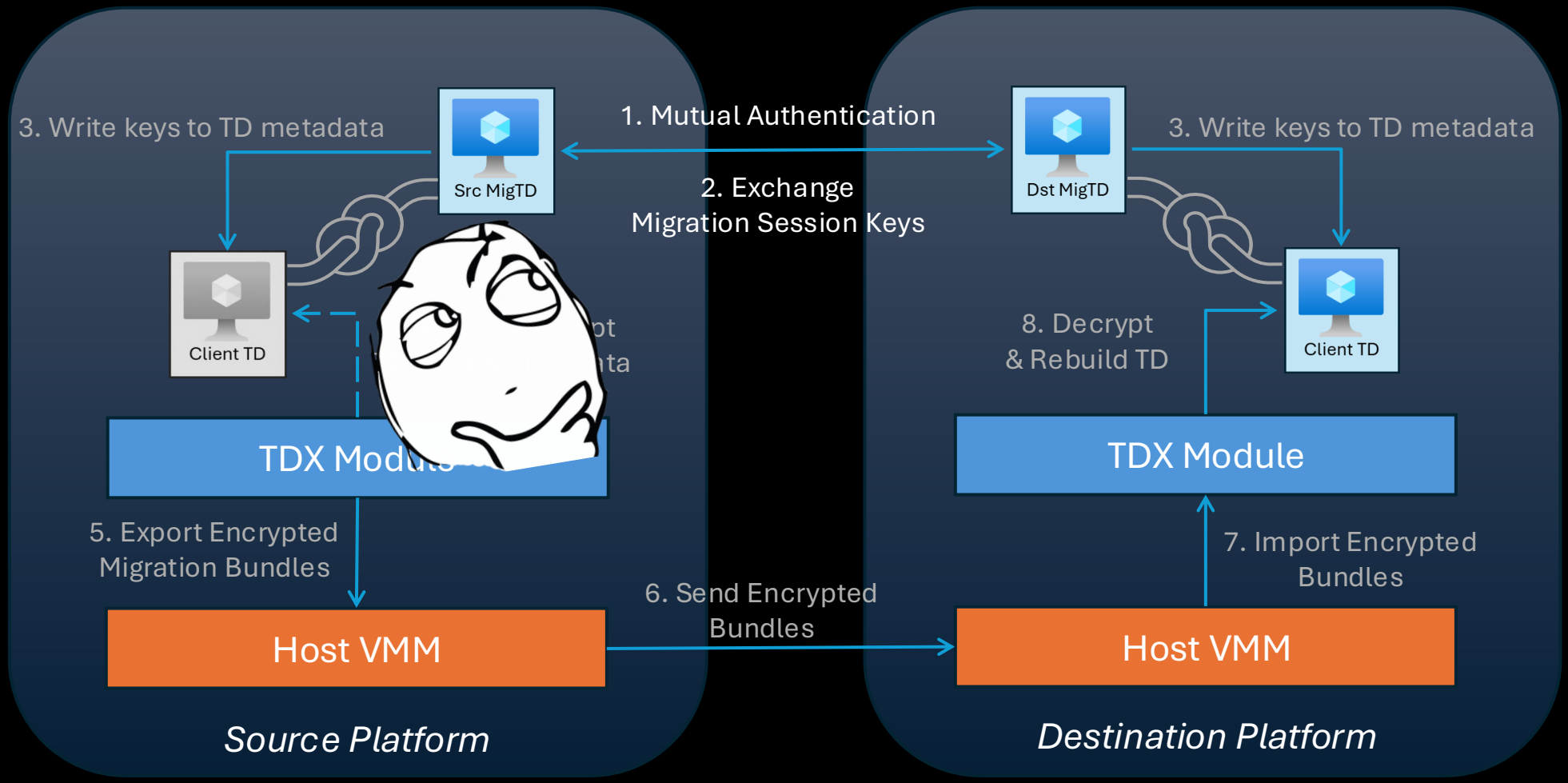
Main TD

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD's ATTRIBU
    uint64_t        xfam; /**< TD's XFAM**/
    measurement_t   mr_td; /**< Measurement of th
    /**
     * 48 Software defined ID for additional con
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Software defin
    /**
     * Software defined ID for owner-defined con
     * e.g., specific to the workload rather tha
     */
    measurement_t   mr_owner_config;
    measurement_t   rtmr[NUM_OF_RTMRS]; /**<  Arr
    measurement_t   servtd_hash;

    uint8_t         reserved[64];
} td_info_t;
```

Live Migration Flow (high level)



Live Migration Flow (high level)

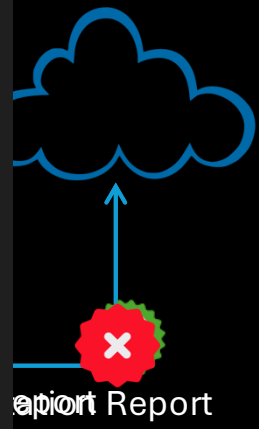


“What if the Migration TD isn’t trusted?”

Slides to CVE

Host & Migration TD join forces

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD's ATTRIBUTES */
    uint64_t        xfam; /**< TD's XFAM**/
    measurement_t   mr_td; /**< Measurement of the initial contents of the TD */
    /**
     * 48 Software defined ID for additional configuration for the software in the TD
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Software defined ID for TD's owner */
    /**
     * Software defined ID for owner-defined configuration of the guest TD,
     * e.g., specific to the workload rather than the runtime or OS.
     */
    measurement_t   mr_owner_config;
    measurement_t   rtmr[NUM_OF_RTMRS]; /**<  Array of NUM_RTMRS runtime extendable measurement registers */
    measurement_t   servtd_hash;

    uint8_t         reserved[64];
} td_info_t;
```



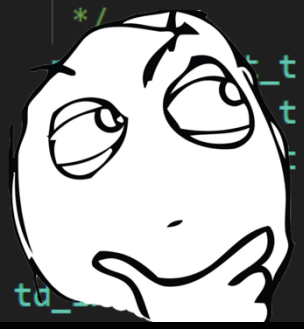
"Ok, so we just won't use migratable TDs"

Migration attribute

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD's ATTR
    uint64_t        xfam; /**< TD's XFAM**/
    measurement_t   mr_td; /**< Measurement of
    /**
     * 48 Software defined ID for additional
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Software de
    /**
     * Software de
     * e.g., speci
     */
                                                    sters */
    reserved[64];
} td_
```

```
typedef union td_param_attributes_s {
    struct
    {
        uint64_t debug           : 1;   // Bit 0
        uint64_t reserved_tud    : 7;   // Bits 7:1
        uint64_t reserved_sec    : 20;  // Bits 28:8
        uint64_t sept_ve_disable : 1;   // Bit  28 -
        uint64_t migratable      : 1;   // Bit 29
        uint64_t pks             : 1;   // Bit 30
        uint64_t kl              : 1;   // Bit 31
                                        // Bits 62:32
                                        // Bit 63
```

```
if (!tdcs_p->executions_ctl_fields.attributes.migratable)
{
    return_val = TDX_TD_NOT_MIGRATABLE;
    goto EXIT;
}
```

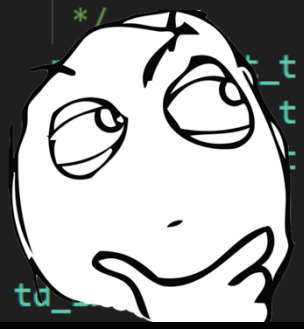


Migration attribute

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes
    uint64_t        xfam; /**<
    measurement_t   mr_td; /**
    /**
     * 48 Software defined ID
     */
    measurement_t   mr_config_
    measurement_t   mr_owner;
    /**
     * Software de
     * e.g., speci
     */
                                    reserved[64];
} td_
```

```
tes_s {
    : 1;    // Bit 0
    : 7;    // Bits 7:1
    : 20;   // Bits 28:8
le  : 1;    // Bit  28 -
    : 1;    // Bit 29
    : 1;    // Bit 30
    : 1;    // Bit 31
gratable)   // Bits 62:32
            // Bit 63
sters */
```

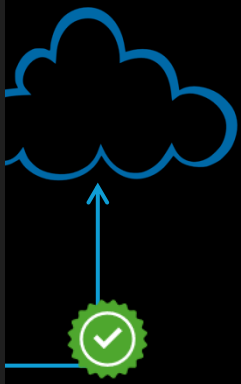
```
if (!tdcs
{
    retur
    goto
}
```



Host & Migration TD join forces

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD'
    uint64_t        xfam; /**< TD's XFAM
    measurement_t   mr_td; /**< Measurem
    /**
     * 48 Software defined ID for addit
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Softw
    /**
     * Software defined ID for owner-de
     * e.g., specific to the workload r
     */
    measurement_t   mr_owner_config;
    measurement_t   rtmr[NUM_OF_RTMRS]; /**<  Array of NUM_RTMRS runtime extendable measurement registers */
    measurement_t   servtd_hash;

    uint8_t         reserved[64];
} td_info_t;
```

```
typedef union td_param_attributes_s {
    struct
    {
        uint64_t debug           : 1;   // Bit 0
        uint64_t reserved_tud    : 7;   // Bits 7:1
        uint64_t reserved_sec    : 20;  // Bits 28:8
        uint64_t sept_ve_disable : 1;   // Bit  28 -
        uint64_t migratable      : 1;   // Bit 29
        uint64_t pks             : 1;   // Bit 30
        uint64_t kl              : 1;   // Bit 31
        uint64_t reserved_other  : 31;  // Bits 62:32
        uint64_t perfmon         : 1;   // Bit 63
    };
    uint64_t raw;
} td_param_attributes_t;
```



6. Decrypt, *Manipulate*, Encrypt

Result

- Customers can’t distinguish between a fresh valid TD, and a rooted one.

→ Goal #1 ✅

Intel fixes

1. Added a check for migratable bit on import

No CVE (fixed for TDX 1.5.01.02 release)

```
if (!tdcs_p->executions_ctl_fields.attributes.migratable)
{
    return_val = TDX_TD_NOT_MIGRATABLE;
    goto EXIT;
}
```

Intel fixes

2. Bound destination TD hash into attestation report - CVE-2023-45745

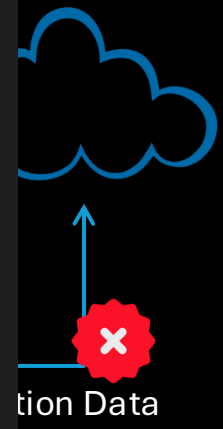
```
{
  // SERVTD_HASH // 72
  .field_id =  { .raw  = 0x9910000300000000 },
  .num_of_fields = 1, .num_of_elem = 6, .offset = 0x0D00, .attributes = { .raw = 0x0 },
  .prod_rd_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .prod_wr_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .dbg_rd_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .dbg_wr_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .guest_rd_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .guest_wr_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .migtd_rd_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .migtd_wr_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .export_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .import_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL),
  .special_rd_handling = false, .special_wr_handling = false,
  .mig_export = MIG_MB, .mig_import = MIG_MB
},
```

change

```
{
  // SERVTD_HASH // 75
  .field_id =  { .raw  = 0x9910000300000000 },
  .num_of_fields = 1, .num_of_elem = 6, .offset = 0x0D00, .attributes = { .raw = 0x0 },
  .prod_rd_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .prod_wr_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .dbg_rd_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .dbg_wr_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .guest_rd_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .guest_wr_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .migtd_rd_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .migtd_wr_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .export_mask = (-1ULL & 0xFFFFFFFFFFFFFFFFULL), .import_mask = (0ULL & 0xFFFFFFFFFFFFFFFFULL),
  .special_rd_handling = false, .special_wr_handling = false,
  .mig_export = MIG_MB, .mig_import = MIG_IBS
},
```

Destination TD hash is immutable

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD's ATTRIBUTES */
    uint64_t        xfam; /**< TD's XFAM**/
    measurement_t   mr_td; /**< Measurement of the initial contents of the TD */
    /**
     * 48 Software defined ID for additional configuration for the software in the TD
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Software defined ID for TD's owner */
    /**
     * Software defined ID for owner-defined configuration of the guest TD,
     * e.g., specific to the workload rather than the runtime or OS.
     */
    measurement_t   mr_owner_config;
    measurement_t   rtmr[NUM_OF_RTMRS]; /**<  Array of NUM_RTMRS runtime extendable measurement registers */
    measurement_t   servtd_hash;

    uint8_t         reserved[64];
} td_info_t;
```



Source Platform

Destination Platform

New goal:
Hijack a Running TD

The method: Instance binding

Instance Binding

- Every privileged operation of a Service TD on its target triggers a TD_INFO hash check.
- If any value in the TD_INFO changes, the binding will break
- In such a case, the host can rebind the Service TD using **Instance Binding.**

Service TD

```
typedef struct PACKED td_info_s
{
    uint64_t        attributes; /**< TD's ATTRIBU
    uint64_t        xfam; /**< TD's XFAM**/
    measurement_t   mr_td; /**< Measurement of th
    /**
     * 48 Software defined ID for additional con
     */
    measurement_t   mr_config_id;
    measurement_t   mr_owner; /**< Software defin
    /**
     * Software defined ID for owner-defined con
     * e.g., specific to the workload rather tha
     */
    measurement_t   mr_owner_config;
    measurement_t   rtmr[NUM_OF_RTMRS]; /**<  Arr
    measurement_t   servtd_hash;

    uint8_t         reserved[64];
} td_info_t;
```

Binding Table

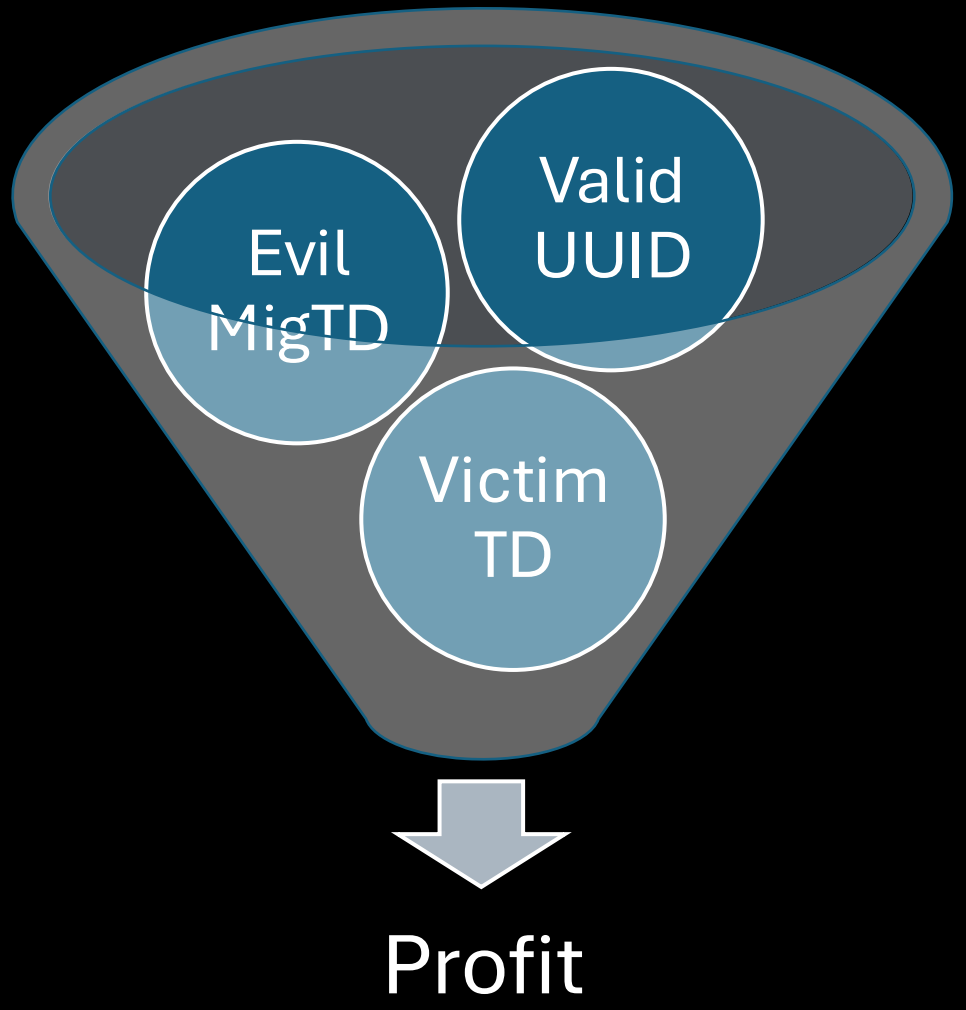
UUID	Service TD unique identifier
INFO HASH	Hash of the Service TD's TD_INFO
…	…

Instance Binding

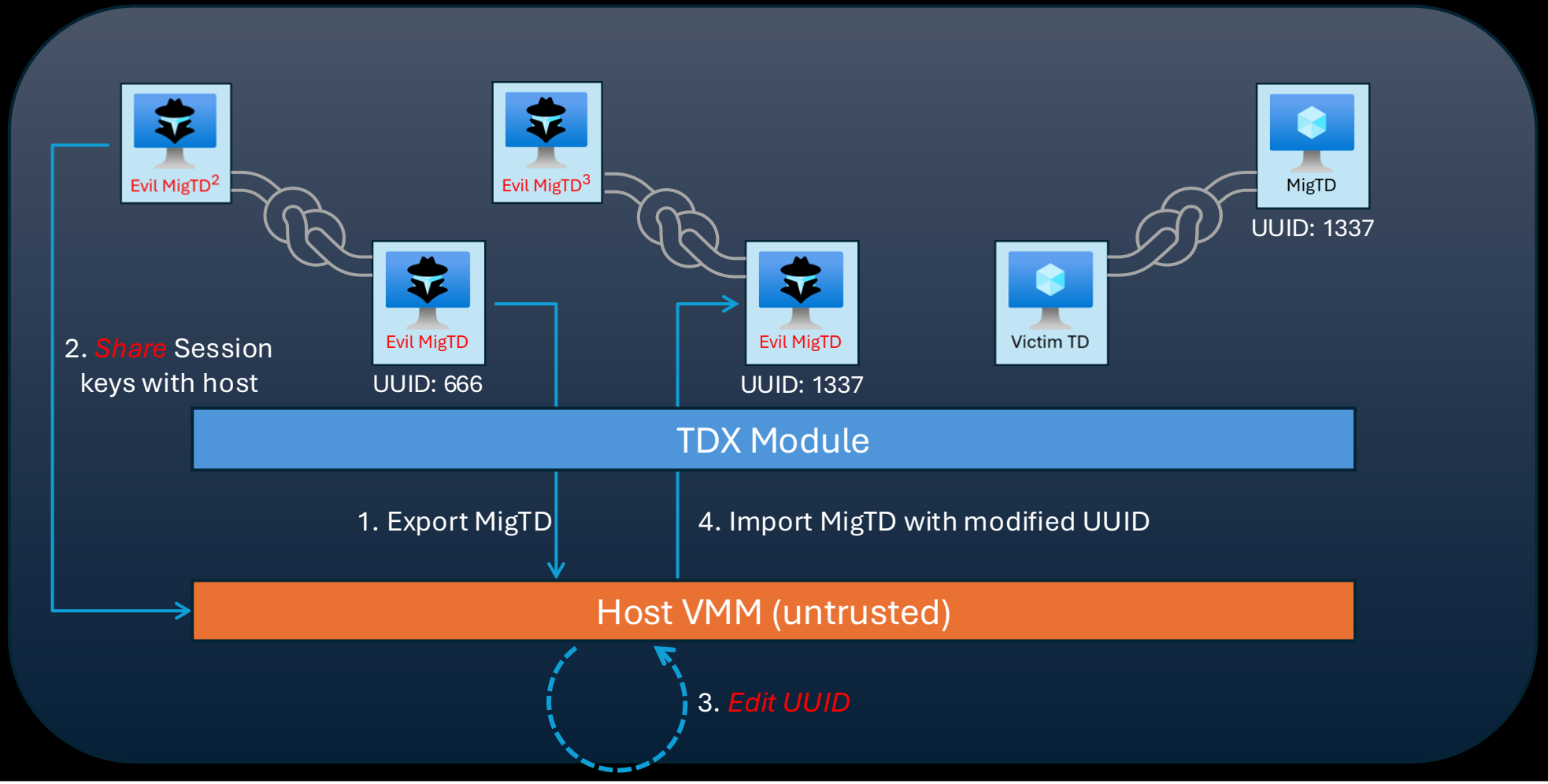
```
if (servtd_attr.instance_binding == 1)
{
    if (!tdx_memcmp(&tdcs_p->service_td_fields.servtd_bindings_table[servtd_slot].uuid,
            &servtd_tdr_p->management_fields.td_uuid, sizeof(servtd_tdr_p->management_fields.td_uuid)))
    {
        return_val = TDX_SERVTD_UUID_MISMATCH;
        goto EXIT;
    }
    tdx_memcpy(tdcs_p->service_td_fields.servtd_bindings_table[servtd_slot].info_hash.qwords, sizeof(measurement_t),
                servtd_info_hash.qwords, sizeof(servtd_info_hash));

}
```

Instance Binding – Mission Plan

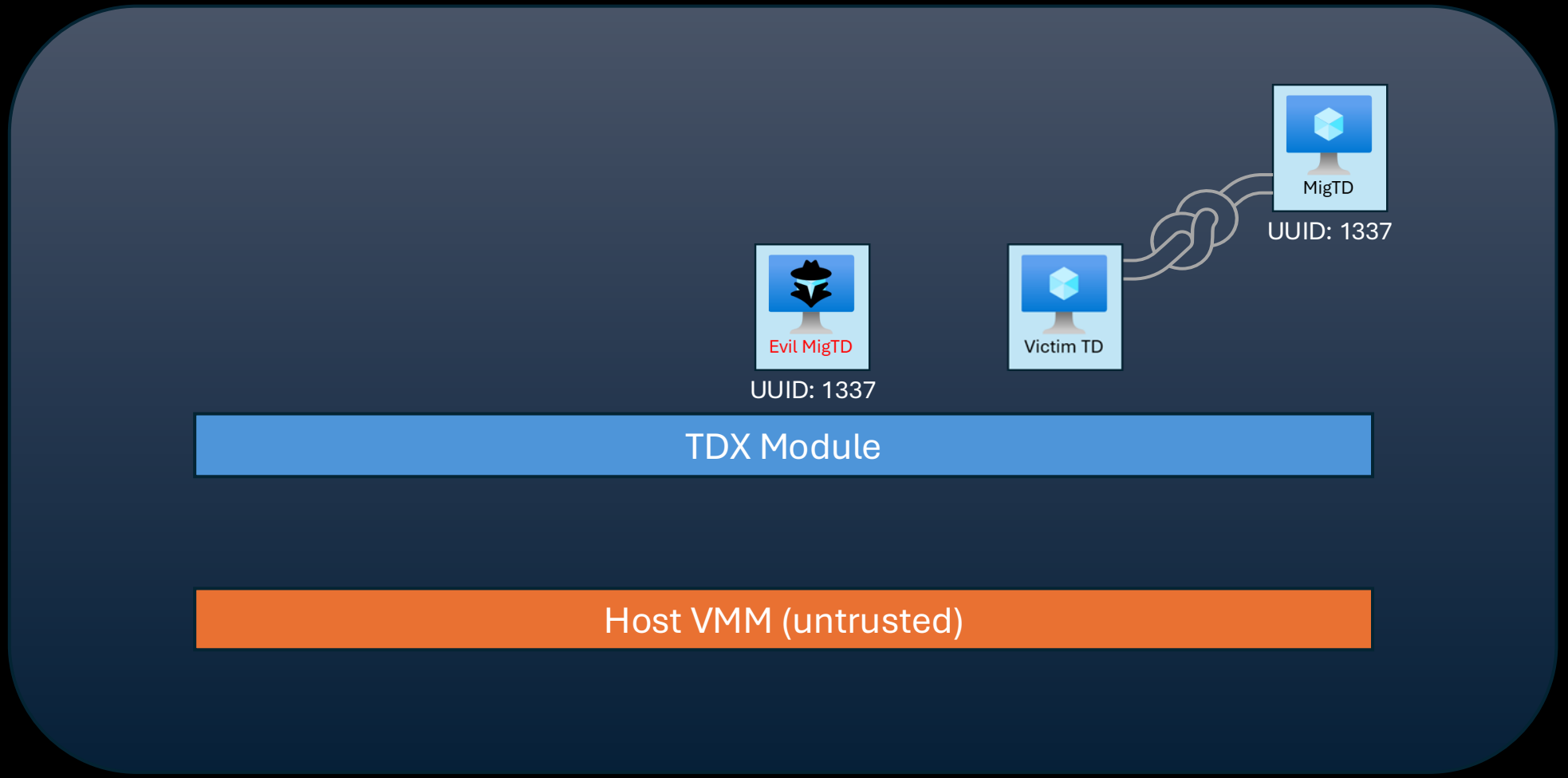
- Fake an evil MigTD's UUID
- "Rebind" evil MigTD to a victim TD
- Take over the victim TD



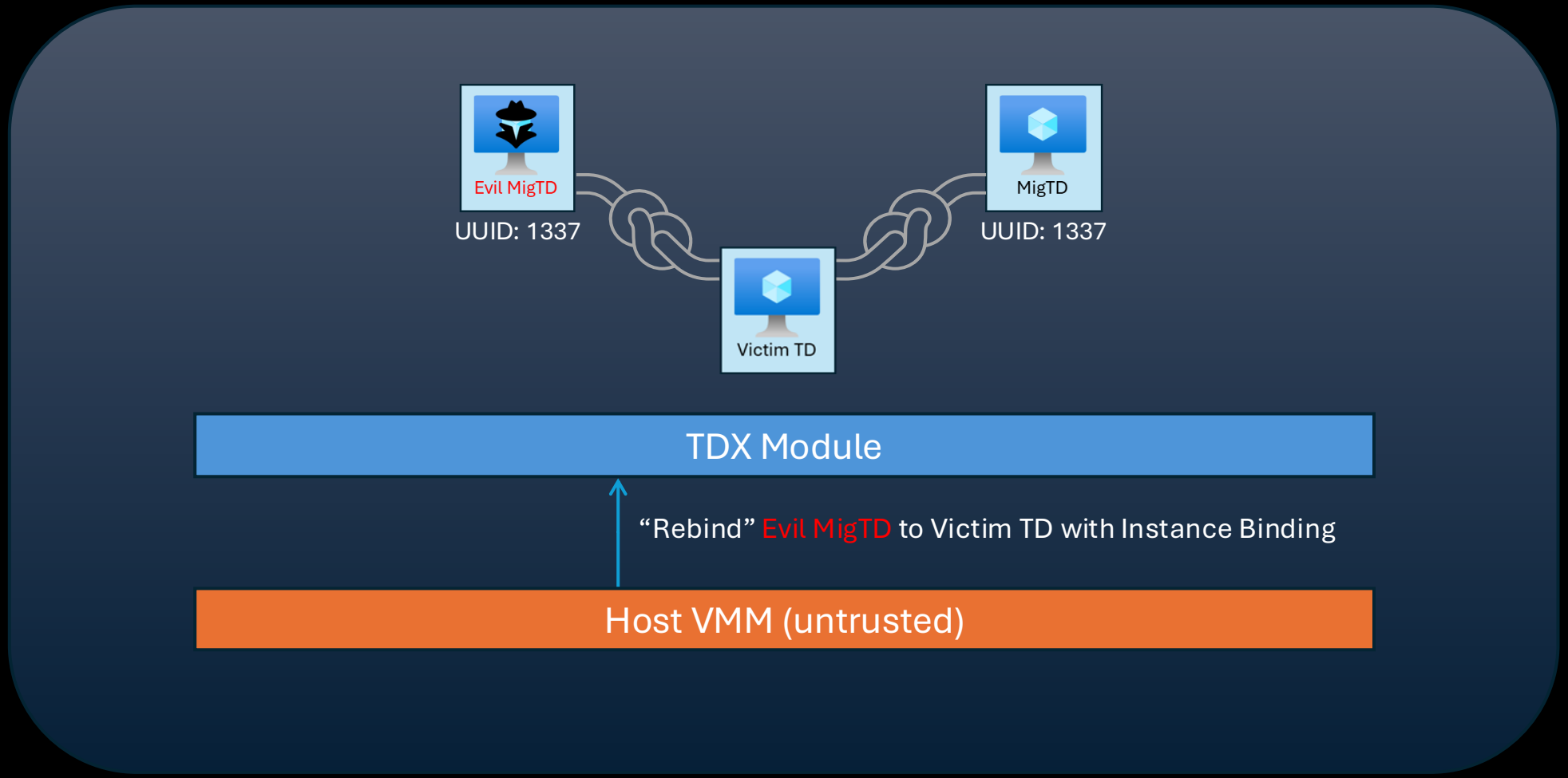
Instance Binding – Fake the UUID



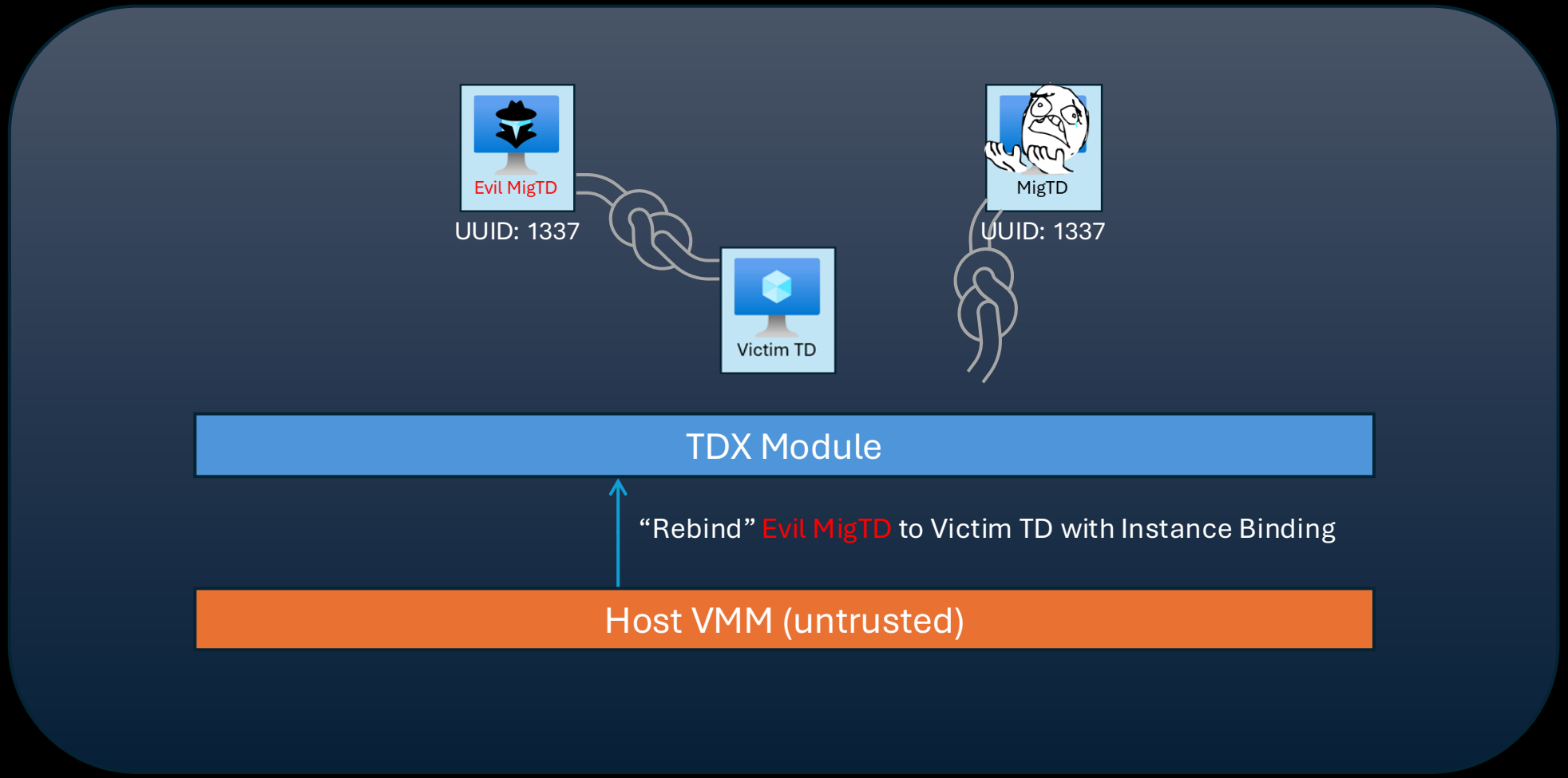
Instance Binding – Fake the UUID



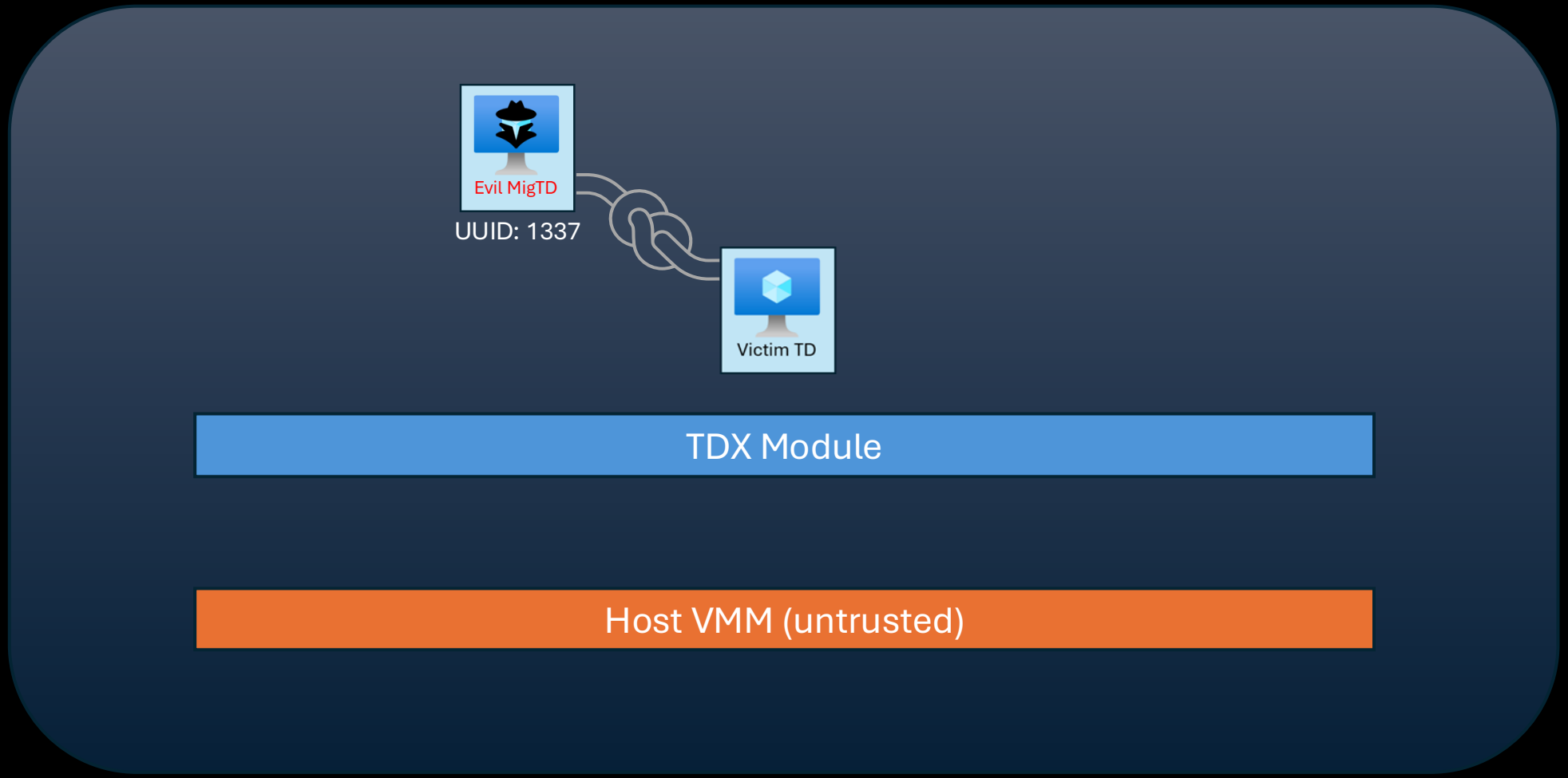
Instance Binding – Rebind



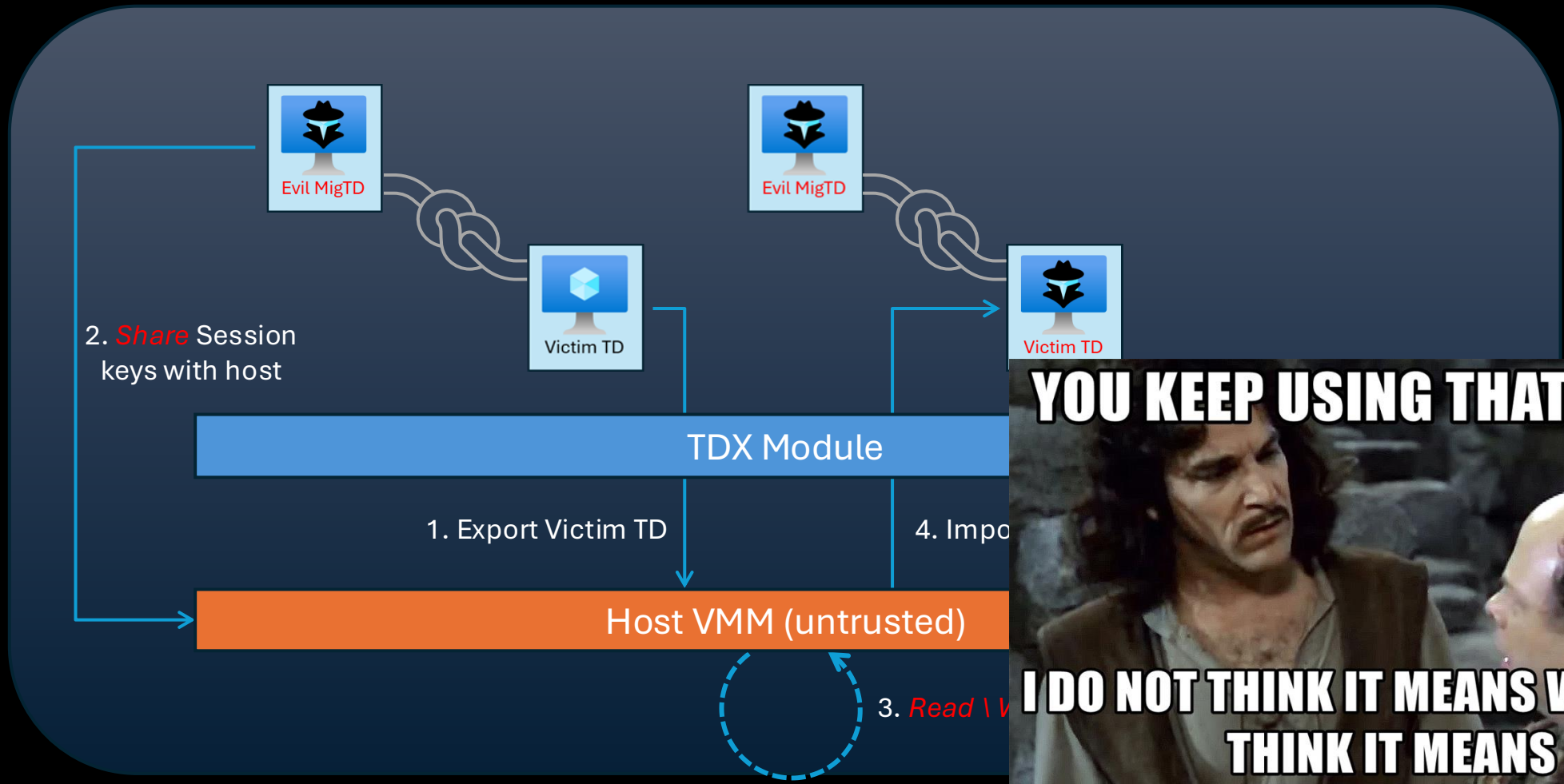
Instance Binding – Rebind



Instance Binding – Rebind



Instance Binding – TD Takeover



Intel fixes

- Added a check for migratable bit on import – no CVE (fixed before release)
- Bound destination TD hash into attestation report

  CVE-2023-45745
- Disable Instance Binding feature

  CVE-2023-47855

End result

- Customers can’t distinguish between a fresh valid TD, and a rooted one.

→ Goal #1 ✅

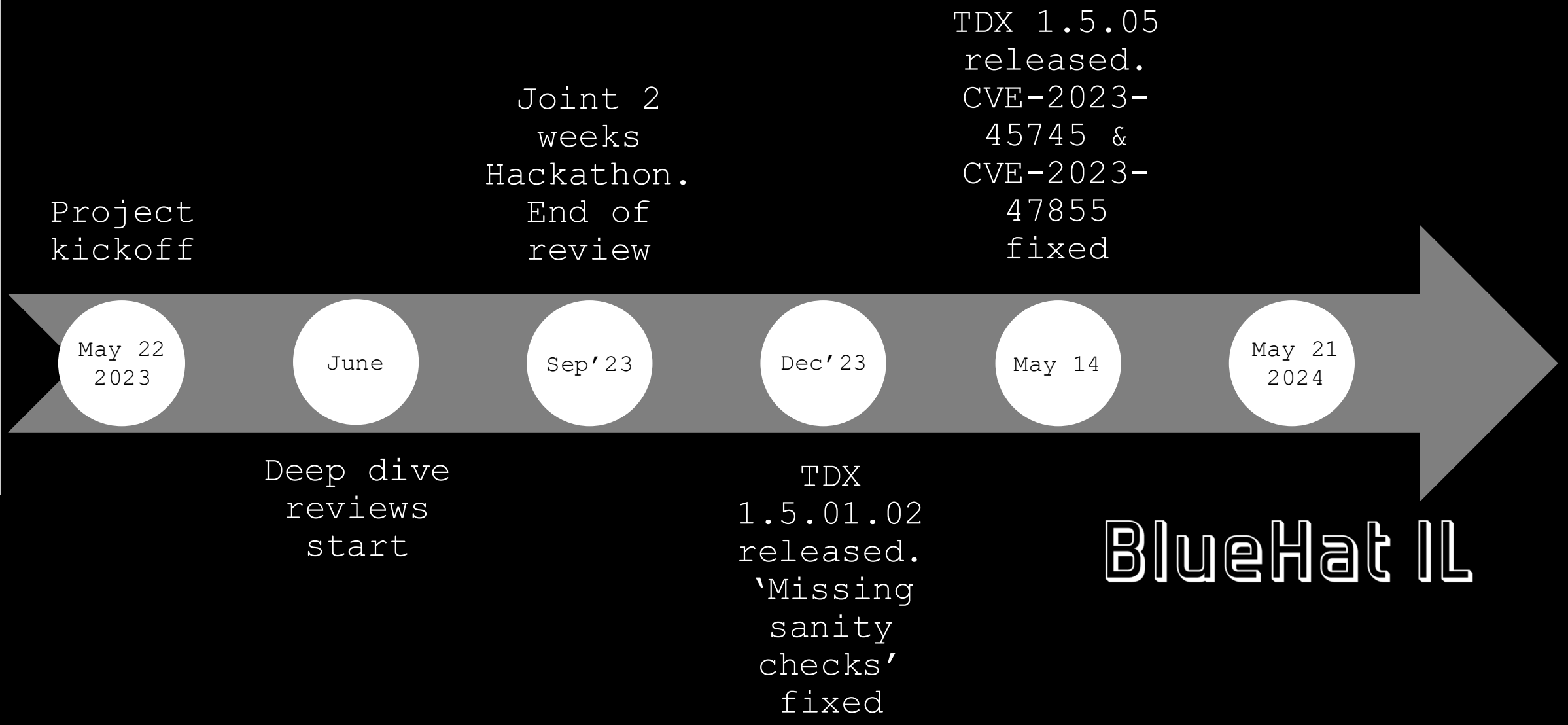
- Even existing Migratable TDs can be compromised, and secrets stolen

→ Goal #2 ✅

Or in other words…


CATS: ALL YOUR TD ARE BELONG
TO US.

Timelines



BlueHat IL

Summary

- In total we had 21 findings, with 6 confirmed vulnerabilities – out of which we presented 2.5 today. Some additional findings will be disclosed in the coming months.
- White paper covering all the research is coming shortly.
- Intel TDX 1.5 was clearly written with security in mind, and finding bugs was hard. Most vulnerabilities were found around gaps in the threat model between both companies, and this gap is now closed.
- Our confidence in confidential compute maturity raised significantly following this review.

Happy hour time!

Thank you for listening