Manual Approval Transfer Manager

- **Introduced in:** 2.0.0
- **Contract name:** ManualApprovalTransferManager.sol
- **Type:** Transfer Manager Module
- **Compatible Protocol Version:** TBD
- **Associated LucidChart**: TBD
- **Github Repository**: https://github.com/PolymathNetwork/polymath-core/blob/master/contracts/modules/TransferManager/ManualApprovalTransferManager.sol

How it works

The Manual Approval Transfer Manager module works by allowing to manually approve or block transactions between account addresses. The Manual approval allows the issuer to create an allowance (that has been approved) with an added expiry time frame for the allowance period. **This will cause a transaction from said from/to accounts to succeed even if a different TM said otherwise (or even if those accounts were not part of the GTM's whitelist).**
Whereas the manual blocking allows the issuer to specify a list of blocked address pairs with an associated expiry time for the block. **In this case, this will make the transaction fail, even if all the other TMs said the transaction was approved.**

Key functionalities (as defined in the Smart Contract)

Get Init Function

This module has no initialization.

Verify Transfer

Summary: This function is used to verify the transfer transaction and allow a manually approved transaction to bypass other restrictions

Note: This function must only be called by the associated security token.

```
/**
     * @notice Used to verify the transfer transaction and all
ow a manually approved transqaction to bypass other restrictio
ns
     * @param _from Address of the sender
     * @param _to Address of the receiver
     * @param _amount The amount of tokens to transfer
     */
    function executeTransfer(
        address _from,
        address _to,
        uint256 _amount,
        bytes calldata /* _data */
    )
        external
        onlySecurityToken
        returns(Result)
```

Add Manual Approval

Summary: This function allows the issuer to add a pair of addresses to the manual approvals.
Note: An investor can only have one manual approval at a time. If a second manual approval needs to be added for an investor, the existing one must first be revoked unless it has a zero allowance.

```
/**
    * @notice Adds a pair of addresses to manual approvals
    * @param _from is the address from which transfers are app
roved
```

```
    * @param _to is the address to which transfers are approve
d
    * @param _allowance is the approved amount of tokens
    * @param _expiryTime is the time until which the transfer
is allowed
    * @param _description Description about the manual approva
l
    */
    function addManualApproval(
        address _from,
        address _to,
        uint256 _allowance,
        uint256 _expiryTime,
        bytes32 _description
    )
        external
        withPerm(ADMIN)
```

Add Manual Approval Multi

Summary: This function allows the issuer to add multiple manual approvals in a batch.

```
/**
    * @notice Adds mutiple manual approvals in batch
    * @param _from is the address array from which transfers a
re approved
    * @param _to is the address array to which transfers are a
pproved
    * @param _allowances is the array of approved amounts
```

```
    * @param _expiryTimes is the array of the times until whic
h eath transfer is allowed
    * @param _descriptions is the description array for these
manual approvals
    */
    function addManualApprovalMulti(
        address[] memory _from,
        address[] memory _to,
        uint256[] memory _allowances,
        uint256[] memory _expiryTimes,
        bytes32[] memory _descriptions
    )
        public
        withPerm(ADMIN)
```

Modify Manual Approval

Summary: This function allows the issuer to modify their existing manual approvals.

```
/**
    * @notice Modify the existing manual approvals
    * @param _from is the address from which transfers are app
roved
    * @param _to is the address to which transfers are approve
d
    * @param _expiryTime is the time until which the transfer
is allowed
    * @param _changeInAllowance is the change in allowance
    * @param _description Description about the manual approva
l
```

```
    * @param _increase tells whether the allowances will be in
creased (true) or decreased (false).
    * or any value when there is no change in allowances
    */
    function modifyManualApproval(
        address _from,
        address _to,
        uint256 _expiryTime,
        uint256 _changeInAllowance,
        bytes32 _description,
        bool _increase
    )
        external
        withPerm(ADMIN)
```

Modify Manual Approval Multi

Summary: This function allows the issuer to add multiple manual approvals in a batch.

```
/**
     * @notice Adds mutiple manual approvals in batch
     * @param _from is the address array from which transfers
are approved
     * @param _to is the address array to which transfers are
approved
     * @param _expiryTimes is the array of the times until whi
ch eath transfer is allowed
     * @param _changeInAllowance is the array of change in all
owances
     * @param _descriptions is the description array for these
manual approvals
```

```
     * @param _increase Array of bools that tells whether the
allowances will be increased (true) or decreased (false).
     * or any value when there is no change in allowances
     */
    function modifyManualApprovalMulti(
        address[] memory _from,
        address[] memory _to,
        uint256[] memory _expiryTimes,
        uint256[] memory _changeInAllowance,
        bytes32[] memory _descriptions,
        bool[] memory _increase
    )
        public
        withPerm(ADMIN)
```

Revoke Manual Approval

Summary: This function allows the issuer to remove addresses from the manual approvals. For example, when adding a manual approval, an “entry” is created saying A to B to allow this transfer.
This function deletes said entry.

```
/**
    * @notice Removes a pairs of addresses from manual approva
ls
    * @param _from is the address from which transfers are app
roved
    * @param _to is the address to which transfers are approve
d
    */
    function revokeManualApproval(address _from, address _to)
external withPerm(ADMIN)
```

Revoke Manual Approval Multi

Summary: This function allows the issuer to remove multiple pairs of addresses form manual approvals.
Requirements:

- This function requires that (_from.length == _to.length, "Input array length mismatch")

```
/**
    * @notice Removes mutiple pairs of addresses from manual approvals
    * @param _from is the address array from which transfers are approved
    * @param _to is the address array to which transfers are approved
    */
    function revokeManualApprovalMulti(address[] calldata _from, address[] calldata _to) external withPerm(ADMIN)
```

Get Active Approvals To User

Summary: This function gets called to return all the actives approvals that correspond to a specific address.

```
/**
     * @notice Returns the all active approvals corresponds to an address
     * @param _user Address of the holder corresponds to whom list of manual approvals
     * need to return
     * @return address[] addresses from
     * @return address[] addresses to
     * @return uint256[] allowances provided to the approvals
```

```
     * @return uint256[] expiry times provided to the approval
s
     * @return bytes32[] descriptions provided to the approval
s
     */
    function getActiveApprovalsToUser(address _user) external
view returns(address[] memory, address[] memory, uint256[] mem
ory, uint256[] memory, bytes32[] memory)
```

Get Approval Details

Summary: This function retrieves the details of the approval that corresponds to **_from** and **_to** addresses.

```
/**
     * @notice Get the details of the approval corresponds to
_from & _to addresses
     * @param _from Address of the sender
     * @param _to Address of the receiver
     * @return uint256 expiryTime of the approval
     * @return uint256 allowance provided to the approval
     * @return uint256 Description provided to the approval
     */
    function getApprovalDetails(address _from, address _to) ex
ternal view returns(uint256, uint256, bytes32)
```

Get Total Approvals Length

Summary: This function simply returns the current number of active approvals

```
/**
* @notice Returns the current number of active approvals
```

```
*/
    function getTotalApprovalsLength()
}
```

Get All Approvals

Summary: Get the details of all the transfer approvals

```
/**
     * @notice Get the details of all approvals
     * @return address[] addresses from
     * @return address[] addresses to
     * @return uint256[] allowances provided to the approvals
     * @return uint256[] expiry times provided to the approval
s
     * @return bytes32[] descriptions provided to the approval
s
     */
    function getAllApprovals() external view returns(address[]
memory, address[] memory, uint256[] memory, uint256[] memory,
bytes32[] memory)
```

Special considerations / notes

None

Troubleshooting / FAQs

None

Know Issues / bugs

- If a manual approval exists for two addresses, the only way to add a new manual approval for those address is revoking the first one, even if it was expired. (This only affects further approvals between the SAME addresses in the SAME

direction. I.E: If A → B = 100 tokens, can't create a new A → B = 500 until the first A →B has been revoked. But A → C, B → C, C → A, C → B, and even B → A are still possible).
    - **Scope:** Address-pair level, in the same order i.e. if allow 100 tokens from A to B, you can allow 1000 tokens from B to A. You will need to revoke the first approval if you want to allow 1000 tokens from A to B, even if the first approval is expired.
- It is possible to add a manual approval and a manual blocking for the same pair of addresses. Due to the logic on verify transfer, blocking will always win.

Changelog

For 2.0.1 (November / December 2018 - Not release yet):

- **Removed:** `0x0` check for the `_from` address to `ManualApprovalTransferManager`. This allows for the Issuer/Transfer Agent to approve a one-off mint of tokens that otherwise would not be possible.
- **Changed:** The version of `ManualApprovalTransferManagerFactory` from `1.0.0` to `2.0.1`.
- **Deployed:** 2.0.1 `ManualApprovalTransferManagerFactory` to address 0x6af2afad53cb334e62b90ddbdcf3a086f654c298
- **Added:** `getActiveApprovalsToUser()` function to access all the active approvals for a user whether user is in the `from` or in `to`.
- **Added:** `getApprovalDetails()` to get the details of the approval corresponds to `_from` and `_to` address.
- **Added:** feature to modify the details of the active approval using `modifyApproval()` & `modifyApprovalMulti()`.
- **Added:** `addManualApprovalMulti()` and `revokeManualApprovalMulti()` batch function for adding and revoking the manual approval respectively.