MythX

REPORT 628D6808666C2300183596D9

Created	Tue May 24 2022 23:19:36 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	6197960e3494e9c8c076e89b

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
66757573-d200-4fcc-bb5b-6cdbe21cbea7	RewardsDistributor.sol	0

Analysis 66757573-d200-4fcc-bb5b-6cdbe21cbea7 MythX

Started	Tue May 24 2022 23:19:40 GMT+0000 (Coordinated Universal Time)
Finished	Tue May 24 2022 23:21:50 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Remythx
Main Source File	RewardsDistributor.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	0

ISSUES

UNKNOWN Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

RewardsDistributor.sol

Locations

```
);

uint constant WEEK = 7 * 86400;

uint public start_time;
```

UNKNOWN Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

RewardsDistributor.sol

Locations

```

constructor(address _voting_escrow) {
uint _t = block.timestamp / WEEK * WEEK;
start_time = _t;
last_token_time = _t;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```

constructor(address _voting_escrow) {
uint _t = block.timestamp / WEEK * WEEK;
start_time = _t;
last_token_time = _t;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```

function timestamp() external view returns (uint) {
return block.timestamp / WEEK * WEEK;
}

```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```

function timestamp() external view returns (uint) {
return block.timestamp / WEEK * WEEK;
}

```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
function _checkpoint_token() internal {
uint token_balance = IERC20(token).balanceOf(address(this));
uint to_distribute = token_balance - token_last_balance;
token_last_balance = token_balance;

```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```

uint t = last_token_time;
uint since_last = block.timestamp - t;
last_token_time = block.timestamp;
uint this_week = t / WEEK * WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
uint since_last = block.timestamp - t;
last_token_time = block.timestamp;
uint this_week = t / WEEK * WEEK;
uint next_week = 0;

```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
uint since_last = block.timestamp - t;
last_token_time = block.timestamp;
uint this_week = t / WEEK * WEEK;
uint next_week = 0;

```

UNKNOWN

SWC-101

Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
uint next_week = 0;

for (uint i = 0; i < 20; i++) {
next_week = this_week + WEEK;
if (block.timestamp < next_week) {
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```

for (uint i = 0; i < 20; i++) {
next_week = this_week + WEEK;
if (block.timestamp < next_week) {
if (since_last == 0 && block.timestamp == t) {
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
if (block.timestamp < next_week) {
if (since_last == 0 && block.timestamp == t) {
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (block.timestamp - t) / since_last;
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (block.timestamp - t) / since_last;
}
break;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (block.timestamp - t) / since_last;
}
break;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (block.timestamp - t) / since_last;
}
break;
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (block.timestamp - t) / since_last;
}
break;
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
} else {
if (since_last == 0 && next_week == t) {
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (next_week - t) / since_last;
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (next_week - t) / since_last;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (next_week - t) / since_last;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (next_week - t) / since_last;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (next_week - t) / since_last;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
uint _min = 0;
uint _max = IVotingEscrow(ve).epoch();
for (uint i = 0; i < 128; i++) {
if (_min >= _max) break;
uint _mid = (_min + _max + 2) / 2;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
for (uint i = 0; i < 128; i++) {
if (_min >= _max) break;
uint _mid = (_min + _max + 2) / 2;
IVotingEscrow.Point memory pt = IVotingEscrow(ve).point_history(_mid);
if (pt.ts <= _timestamp) {
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
for (uint i = 0; i < 128; i++) {
if (_min >= _max) break;
uint _mid = (_min + _max + 2) / 2;
IVotingEscrow.Point memory pt = IVotingEscrow(ve).point_history(_mid);
if (pt.ts <= _timestamp) {
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
for (uint i = 0; i < 128; i++) {
if (_min >= _max) break;
uint _mid = (_min + _max + 2) / 2;
IVotingEscrow.Point memory pt = IVotingEscrow(ve).point_history(_mid);
if (pt.ts <= _timestamp) {
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
_min = _mid;
} else {
_max = _mid - 1;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
uint _min = 0;
uint _max = max_user_epoch;
for (uint i = 0; i < 128; i++) {
if (_min >= _max) break;
uint _mid = (_min + _max + 2) / 2;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
for (uint i = 0; i < 128; i++) {
if (_min >= _max) break;
uint _mid = (_min + _max + 2) / 2;
IVotingEscrow.Point memory pt = IVotingEscrow(ve).user_point_history(tokenId, _mid);
if (pt.ts <= _timestamp) {
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
for (uint i = 0; i < 128; i++) {
if (_min >= _max) break;
uint _mid = (_min + _max + 2) / 2;
IVotingEscrow.Point memory pt = IVotingEscrow(ve).user_point_history(tokenId, _mid);
if (pt.ts <= _timestamp) {
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
for (uint i = 0; i < 128; i++) {
if (_min >= _max) break;
uint _mid = (_min + _max + 2) / 2;
IVotingEscrow.Point memory pt = IVotingEscrow(ve).user_point_history(tokenId, _mid);
if (pt.ts <= _timestamp) {
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
_min = _mid;
} else {
_max = _mid -1;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
uint epoch = _find_timestamp_user_epoch(ve, _tokenId, _timestamp, max_user_epoch);
IVotingEscrow.Point memory pt = IVotingEscrow(ve).user_point_history(_tokenId, epoch);
return Math.max(uint(int256(pt.bias - pt.slope * (int128(int256(_timestamp - pt.ts))))), 0);
}

```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
uint epoch = _find_timestamp_user_epoch(ve, _tokenId, _timestamp, max_user_epoch);
IVotingEscrow.Point memory pt = IVotingEscrow(ve).user_point_history(_tokenId, epoch);
return Math.max(uint(int256(pt.bias - pt.slope * (int128(int256(_timestamp - pt.ts))))), 0);
}

```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
uint epoch = _find_timestamp_user_epoch(ve, _tokenId, _timestamp, max_user_epoch);
IVotingEscrow.Point memory pt = IVotingEscrow(ve).user_point_history(_tokenId, epoch);
return Math.max(uint(int256(pt.bias - pt.slope * (int128(int256(_timestamp - pt.ts))))), 0);
}

```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
address ve = voting_escrow;
uint t = time_cursor;
uint rounded_timestamp = block.timestamp / WEEK * WEEK;
IVotingEscrow(ve).checkpoint();

```

UNKNOWN

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`RewardsDistributor.sol`

Locations

```
address ve = voting_escrow;
uint t = time_cursor;
uint rounded_timestamp = block.timestamp / WEEK * WEEK;
IVotingEscrow(ve).checkpoint();

```

UNKNOWN

Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow(ve).checkpoint();

for (uint i = 0; i < 20; i++) {
if (t > rounded_timestamp) {
break;
```

UNKNOWN

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`RewardsDistributor.sol`

Locations

```
int128 dt = 0;
if (t > pt.ts) {
dt = int128(int256(t - pt.ts));
}
ve_supply[t] = Math.max(uint(int256(pt.bias - pt.slope * dt)), 0);
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
dt = int128(int256(t - pt.ts));
}
ve_supply[t] = Math.max(uint(int256(pt.bias - pt.slope * dt)), 0);
}
t += WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
dt = int128(int256(t - pt.ts));
}
ve_supply[t] = Math.max(uint(int256(pt.bias - pt.slope * dt)), 0);
}
t += WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
ve_supply[t] = Math.max(uint(int256(pt.bias - pt.slope * dt)), 0);
}
t += WEEK;
}
time_cursor = t;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow.Point memory old_user_point;

for (uint i = 0; i < 50; i++) {
if (week_cursor >= _last_token_time) break;

```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```

if (week_cursor >= user_point.ts && user_epoch <= max_user_epoch) {
user_epoch += 1;
old_user_point = user_point;
if (user_epoch > max_user_epoch) {
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
}
} else {
int128 dt = int128(int256(week_cursor - old_user_point.ts));
uint balance_of = Math.max(uint(int256(old_user_point.bias - dt * old_user_point.slope)), 0);
if (balance_of == 0 && user_epoch > max_user_epoch) break;
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
} else {
int128 dt = int128(int256(week_cursor - old_user_point.ts));
uint balance_of = Math.max(uint(int256(old_user_point.bias - dt * old_user_point.slope)), 0);
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
} else {
int128 dt = int128(int256(week_cursor - old_user_point.ts));
uint balance_of = Math.max(uint(int256(old_user_point.bias - dt * old_user_point.slope)), 0);
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
}

user_epoch = Math.min(max_user_epoch, user_epoch - 1);
user_epoch_of[_tokenId] = user_epoch;
time_cursor_of[_tokenId] = week_cursor;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
IVotingEscrow.Point memory old_user_point;

for (uint i = 0; i < 50; i++) {
if (week_cursor >= _last_token_time) break;

```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```

if (week_cursor >= user_point.ts && user_epoch <= max_user_epoch) {
user_epoch += 1;
old_user_point = user_point;
if (user_epoch > max_user_epoch) {
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
}
} else {
int128 dt = int128(int256(week_cursor - old_user_point.ts));
uint balance_of = Math.max(uint(int256(old_user_point.bias - dt * old_user_point.slope)), 0);
if (balance_of == 0 && user_epoch > max_user_epoch) break;
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
} else {
int128 dt = int128(int256(week_cursor - old_user_point.ts));
uint balance_of = Math.max(uint(int256(old_user_point.bias - dt * old_user_point.slope)), 0);
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
} else {
int128 dt = int128(int256(week_cursor - old_user_point.ts));
uint balance_of = Math.max(uint(int256(old_user_point.bias - dt * old_user_point.slope)), 0);
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```

function claimable(uint _tokenId) external view returns (uint) {
uint _last_token_time = last_token_time / WEEK * WEEK;
return _claimable(_tokenId, voting_escrow, _last_token_time);
}
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```

function claimable(uint _tokenId) external view returns (uint) {
uint _last_token_time = last_token_time / WEEK * WEEK;
return _claimable(_tokenId, voting_escrow, _last_token_time);
}
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
if (block.timestamp >= time_cursor) _checkpoint_total_supply();
uint _last_token_time = last_token_time;
_last_token_time = _last_token_time / WEEK * WEEK;
uint amount = _claim(_tokenId, voting_escrow, _last_token_time);
if (amount != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
if (block.timestamp >= time_cursor) _checkpoint_total_supply();
uint _last_token_time = last_token_time;
_last_token_time = _last_token_time / WEEK * WEEK;
uint amount = _claim(_tokenId, voting_escrow, _last_token_time);
if (amount != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "-=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
if (amount != 0) {
IVotingEscrow(voting_escrow).deposit_for(_tokenId, amount);
token_last_balance -= amount;
}
return amount;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
if (block.timestamp >= time_cursor) _checkpoint_total_supply();
uint _last_token_time = last_token_time;
_last_token_time = _last_token_time / WEEK * WEEK;
address _voting_escrow = voting_escrow;
uint total = 0;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
if (block.timestamp >= time_cursor) _checkpoint_total_supply();
uint _last_token_time = last_token_time;
_last_token_time = _last_token_time / WEEK * WEEK;
address _voting_escrow = voting_escrow;
uint total = 0;
```

UNKNOWN

SWC-101

Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
uint total = 0;

for (uint i = 0; i < _tokenIds.length; i++) {
uint _tokenId = _tokenIds[i];
if (_tokenId == 0) break;
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

RewardsDistributor.sol

Locations

```
if (amount != 0) {
IVotingEscrow(_voting_escrow).deposit_for(_tokenId, amount);
total += amount;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "-=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
}
if (total != 0) {
token_last_balance -= total;
}

```

UNKNOWN

SWC-101

Compiler-rewritable "<uint> - 1" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
_min = _mid;
} else {
_max = _mid - 1;
}
}
```

UNKNOWN

SWC-101

Compiler-rewritable "<uint> - 1" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
_min = _mid;
} else {
_max = _mid -1;
}
}
```

UNKNOWN

SWC-101

Compiler-rewritable "<uint> - 1" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Compiler-rewritable "<uint> - 1" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
}

user_epoch = Math.min(max_user_epoch, user_epoch - 1);
user_epoch_of[_tokenId] = user_epoch;
time_cursor_of[_tokenId] = week_cursor;
```

UNKNOWN

SWC-101

Compiler-rewritable "<uint> - 1" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`RewardsDistributor.sol`

Locations

```
IVotingEscrow.Point memory user_point = IVotingEscrow(ve).user_point_history(_tokenId, user_epoch);

if (week_cursor == 0) week_cursor = (user_point.ts + WEEK - 1) / WEEK * WEEK;
if (week_cursor >= last_token_time) return 0;
if (week_cursor < _start_time) week_cursor = _start_time;
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`libraries/Math.sol`

Locations

```
if (y > 3) {
z = y;
uint x = y / 2 + 1;
while (x < z) {
z = x;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`libraries/Math.sol`

Locations

```
if (y > 3) {
z = y;
uint x = y / 2 + 1;
while (x < z) {
z = x;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`libraries/Math.sol`

Locations

```
while (x < z) {
z = x;
x = (y / x + x) / 2;
}
} else if (y != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`libraries/Math.sol`

Locations

```
while (x < z) {
z = x;
x = (y / x + x) / 2;
}
} else if (y != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`libraries/Math.sol`

Locations

```
while (x < z) {
z = x;
x = (y / x + x) / 2;
}
} else if (y != 0) {
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`libraries/Math.sol`

Locations

```
for (uint256 y = 1 << 255; y > 0; y >>= 3) {
x <<= 1;
uint256 z = 3 * x * (x + 1) + 1;
if (n / y >= z) {
n -= y * z;
```

UNKNOWN

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`libraries/Math.sol`

Locations

```
for (uint256 y = 1 << 255; y > 0; y >>= 3) {
x <<= 1;
uint256 z = 3 * x * (x + 1) + 1;
if (n / y >= z) {
n -= y * z;
```

UNKNOWN

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`libraries/Math.sol`

Locations

```
for (uint256 y = 1 << 255; y > 0; y >>= 3) {
x <<= 1;
uint256 z = 3 * x * (x + 1) + 1;
if (n / y >= z) {
n -= y * z;
```

UNKNOWN

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`libraries/Math.sol`

Locations

```
for (uint256 y = 1 << 255; y > 0; y >>= 3) {
x <<= 1;
uint256 z = 3 * x * (x + 1) + 1;
if (n / y >= z) {
n -= y * z;
```

UNKNOWN

Arithmetic operation "/" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`libraries/Math.sol`

Locations

```
x <<= 1;
uint256 z = 3 * x * (x + 1) + 1;
if (n / y >= z) {
n -= y * z;
x += 1;
```

UNKNOWN

Arithmetic operation "-=" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`libraries/Math.sol`

Locations

```
uint256 z = 3 * x * (x + 1) + 1;
if (n / y >= z) {
n -= y * z;
x += 1;
}
```

UNKNOWN

Arithmetic operation "*" discovered

This plugin produces issues to support false positive discovery within MythX.

SWC-101

Source file

`libraries/Math.sol`

Locations

```
uint256 z = 3 * x * (x + 1) + 1;
if (n / y >= z) {
n -= y * z;
x += 1;
}
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`libraries/Math.sol`

Locations

```
if (n / y >= z) {
n -= y * z;
x += 1;
}
}
```

UNKNOWN

SWC-110

Public state variable with array type causing reacheable exception by default.

The public state variable "tokens_per_week" in "RewardsDistributor" contract has type "uint256[1000000000000000]" and can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```

uint public last_token_time;
uint[1000000000000000] public tokens_per_week;

address public voting_escrow;
```

UNKNOWN

SWC-110

Public state variable with array type causing reacheable exception by default.

The public state variable "ve_supply" in "RewardsDistributor" contract has type "uint256[1000000000000000]" and can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```
uint public token_last_balance;

uint[1000000000000000] public ve_supply;

address public depositor;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```
if (block.timestamp < next_week) {
if (since_last == 0 && block.timestamp == t) {
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (block.timestamp - t) / since_last;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (block.timestamp - t) / since_last;
}
break;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```
} else {
if (since_last == 0 && next_week == t) {
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (next_week - t) / since_last;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```
tokens_per_week[this_week] += to_distribute;
} else {
tokens_per_week[this_week] += to_distribute * (next_week - t) / since_last;
}
}
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```
dt = int128(int256(t - pt.ts));
}
ve_supply[t] = Math.max(uint(int256(pt.bias - pt.slope * dt)), 0);
}
t += WEEK;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

RewardsDistributor.sol

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

RewardsDistributor.sol

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

RewardsDistributor.sol

Locations

```
if (balance_of == 0 && user_epoch > max_user_epoch) break;
if (balance_of != 0) {
to_distribute += balance_of * tokens_per_week[week_cursor] / ve_supply[week_cursor];
}
week_cursor += WEEK;
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

`RewardsDistributor.sol`

Locations

```

for (uint i = 0; i < _tokenIds.length; i++) {
uint _tokenId = _tokenIds[i];
if (_tokenId == 0) break;
uint amount = _claim(_tokenId, _voting_escrow, _last_token_time);
```