Decentraland Land Registry Audit

Contracts Under Audit

contracts/land/ILANDRegistry.sol
contracts/land/IMetadataHolder.sol
contracts/land/LANDRegistry.sol
contracts/land/LANDStorage.sol
contracts/upgradable/DelegateProxy.sol
contracts/upgradable/IApplication.sol
contracts/upgradable/LANDProxy.sol
contracts/upgradable/Ownable.sol
contracts/upgradable/OwnableStorage.sol
contracts/upgradable/Proxy.sol
contracts/upgradable/ProxyStorage.sol

Recommendations

It's unnecessary to check for `0xffffffff` interface. Lines 354 to 346 can be removed.
https://github.com/decentraland/erc721/blob/master/contracts/ERC721Base.sol#L354-L356

LandProxy.sol - contract is not necessary. Proxy.sol can be used without being extended.

Proxy.sol - `require(currentContract != 0);` Consider comparing `currentContract` to `address(0)` to be explicit about type.

Proxy.sol - Extends `Ownable` but `Ownable` functionality is never used. Ownable can be removed from `Proxy.sol` and `Ownable.sol` can be removed all together.

Ownable.sol - `bytesToAddress()` is never used or tested. Consider removing this function.

LANDRegistry.sol - `require(x.length > 0); require(x.length == y.length);` pattern repeated 3 times. Consider separating out into a function or modifier.

LANDRegistry.sol - `transferLandMany()` can call `transferLand()`. Repeats same lines of code. Consider separating out into a function.

LANDRegistry.sol - `onlyUpdateAuthorized()` using `tokenId` instead of `assetId` is inconsistent with the rest of the contract. We recommend updating the whole contract to use `_tokenId` for consistency with the current ERC721 standard.

LANDRegistry.sol - `ping()` is never called from any functions. Should all non-constant functions that are used by the general population be calling ping? Otherwise, can it be removed entirely?

DelegateProxy.sol - `isContract()` - `constant` is deprecated. Consider updating to `view`

Unbounded loops

The following functions use unbounded for loops to iterate arrays:
`assignMultipleParcels()`, `ownerOfLandMany()`, `landOf()`, `transferManyLand()` and `updateManyLandData()`
These functions may fail due to the block gas limit if the array being iterated is too large.
We tested `assignMultipleParcels()` and `landOf()` with a block gas limit of 8,000,000. `assignMultipleParcels()` failed when handling 103 parcels or more. `landOf()` failed when handling 216 parcels or more.
Consider adding a warning that these functions may fail when handling larger array sizes. For the constant functions this only the case when they are called from a smart contract's non-constant function.

Sanity Checks

Consider adding checks for non-zero address function parameters across the land and erc721 codebases where 0 addresses should not be accepted.

Consider adding checks for `x.length > 0` and `x.length == y.length` to `assignMultipleParcels()`

Style

Consider using double quotes instead of single quotes in the following locations:

- `land/contracts/Storage.sol` - lines 3, 5, 7, and 9
- `land/contracts/land/IMetadataHolder.sol` - line 3
- `land/contracts/land/LANDRegistry.sol` - lines 185 and 194
- `land/contracts/upgradable/LANDProxy.sol` - lines 3 and 4
- `land/contracts/upgradable/Ownable.sol` - line 3

`erc721/contracts/ERC721Base.sol` - lines 3, 5, 7, 9, 11, 220, 229, 269, 296, and 366
`erc721/contracts/ERC721Enumerable.sol` - lines 3 and 4
`erc721/contracts/ERC721Holder.sol` - line 3
`erc721/contracts/ERC721Metadata.sol` - lines 3 and 4
`erc721/contracts/FullAssetRegistry.sol` - lines 3, 4, and 5

ERC721 Differences

Non-ERC721 functions

Zeppelin's ERC721Basic lists `function exists(uint256 _tokenId) public view returns (bool _exists);` which is commented out in IERC721Base but is also not defined in the ERC721 standard.

IERC721Base defines `function isAuthorized(address operator, uint256 assetId) external view returns (bool);` which is not listed in the ERC721 standard

ERC721 Functions

ERC721: `function balanceOf(address _owner) external view returns (uint256);`
IERC721Base: `function balanceOf(address holder) external view returns (uint256);`
Recommendation: Change `holder` to `_owner`

ERC721: `function ownerOf(uint256 _tokenId) external view returns (address);`
IERC721Base: `function ownerOf(uint256 assetId) external view returns (address);`
Recommendation: Change `assetId` to `_tokenId`

ERC721: `function safeTransferFrom(address _from, address _to, uint256 _tokenId, bytes data) external payable;`
IERC721Base: `function safeTransferFrom(address from, address to, uint256 assetId, bytes userData) external;`
Recommendation: Change `from` to `_from`, `to` to `_to`, `assetId` to `_tokenId`, `userData` to `data`

ERC721: `function safeTransferFrom(address _from, address _to, uint256 _tokenId) external payable;`

IERC721Base: `function safeTransferFrom(address from, address to, uint256 assetId) external;`
Recommendation: Change `from` to `_from`, `to` to `_to`, `assetId` to `_tokenId`

ERC721: `function transferFrom(address _from, address _to, uint256 _tokenId) external payable;`
IERC721Base: `function transferFrom(address from, address to, uint256 assetId) external;`
Recommendation: Change `from` to `_from`, `to` to `_to`, `assetId` to `_tokenId`

ERC721: `function approve(address _approved, uint256 _tokenId) external payable;`
IERC721Base: `function approve(address operator, uint256 assetId) external;`
Recommendation: Change `operator` to `_approved`, `assetId` to `_tokenId`.

ERC721: `function setApprovalForAll(address _operator, bool _approved) external;`
IERC721Base: `function setApprovalForAll(address operator, bool authorized) external;`
Recommendation: Change `operator` to `_operator`, `authorized` to `_approved`.

ERC721: `function getApproved(uint256 _tokenId) external view returns (address);`
IERC721Base: `function getApprovedAddress(uint256 assetId) external view returns (address);`
Recommendation: Add `function getApproved(uint256 _tokenId) external view returns (address);` to support the ERC721 interface and be backwards compatible with contracts and other code calling `getApprovedAddress()`

ERC721: `function isApprovedForAll(address _owner, address _operator) external view returns (bool);`
IERC721Base: `function isApprovedForAll(address operator, address assetOwner) external view returns (bool);`
Recommendation: Not sure on this one. There is no way for both of these to be defined because they have the same function selector. The parameters `operator/_operator` and `_owner/assetOwner` are switched which could definitely lead to problems. Updating the function will break backwards compatibility but is most likely the best option.

ERC721 Events

IERC721Base defines `event Transfer(address indexed from, address indexed to, uint256 indexed assetId, address operator, bytes userData);` but the ERC721 standard defines `event Transfer(address indexed _from, address indexed _to, uint256 indexed _tokenId);`

IERC721Base defines `event Approval(address indexed owner, address indexed operator, uint256 indexed assetId);` but the ERC721 standard defines `event Approval(address indexed _owner, address indexed _approved, uint256 indexed _tokenId);` (Add underscores)

IERC721Base defines `event ApprovalForAll(address indexed operator, address indexed holder, bool authorized);` but the ERC721 standard defines the event `event ApprovalForAll(address indexed _owner, address indexed _operator, bool _approved);`

ERC721TokenReceiver Differences

ERC721TokenReceiver:
`function onERC721Received(address _operator, address _from, uint256 _tokenId, bytes _data) external returns(bytes4);`
IERC721Receiver:
`function onERC721Received(address _oldOwner, uint256 _tokenId, bytes _userData) external returns (bytes4);`
Recommendation: This should be updated to the ERC721TokenReceiver interface and in ERC721Holder.sol should return 0x150b7a02 instead of 0xf0b9e5ba to reflect the new function selector.