

## Smart Contract

The audit was conducted by TECHFUND between 17th to 26th of October. The code used was from master branch available on 17th October via following repositories

<https://github.com/quras-official/quras-blockchain-csharp>

<https://github.com/quras-official/quras-smartcontract-compiler>

## Vulnerabilities

|        |   |
|--------|---|
| High   | 1 |
| Medium | 5 |
| Low    | 1 |
| Note   | 2 |

We found above vulnerabilities in the code that have been described below. We also tried to perform flooding the network, but were unable to achieve any Critical Issue in the same.

## 1) Issue : String comparison Incorrect OPCODE generation

HIGH

```
using Quras.SmartContract.Framework;
using Quras.SmartContract.Framework.Services.Module;
using System;
using System.Numerics;

namespace qtest
{
    0 references
    public class Contract1 : SmartContract
    {
        0 references
        public static bool Main(string operation, object[] args)
        {
            if (operation == "test") Runtime.Log("This is test");
            if (operation != "nottest") Runtime.Log("This is not test");
            return true;
        }
    }
}
```

Generated OPCODES for first condition :

```
23 c3 // PICKITEM
24 04 // ??
25 74 // t
26 65 // e
27 73 // s
28 74 // t
29 87 // equal
30 6c // FROMALTSTACK
31 76 // dup
32 6b // TOALTSTACK
33 52 // PUSH2
34 52 // PUSH2
35 7a // roll
36 c4 // setitem
37 6c // FROMALTSTACK
38 76 // dup
39 6b // TOALTSTACK
40 52 // PUSH2
41 c3 // PICKITEM
42 64 // JMPIFNOT
43 25 // 37
44 00 // |
45 0c // ?
46 54 T
47 68 h
48 69 i
49 73 s
50 20
51 69 i
52 73 s
53 20
54 74 t
55 65 e
56 73 s
57 74 t
58 61 // NOP
59 68 //syscall
```

The jump here is correct to FORMALTSTACK for comparison with first operation

Generated OPCODES for second condition :

```

5  c3
6  64 // JMPIFNOT
7  29
8  00
9  10
0  54 T
1  68 h
2  69 i
3  73 s
4  20
5  69 i
6  73 s
7  20
8  6e n
9  6f o
0  74 t
1  20
2  74 t
3  65 e
4  73 s
5  74 t
6  61 // NOP
7  60 // syscall

```

JMPIFNOT jumps to a location which is not valid.

Hence the output is as follows :

Expected output for operation="test" :

**"This is test"**

**"This is not test"**

Output :

**"This is test"**

## Reason :

Invalid else condition

([https://github.com/quras-official/quras-smartcontract-compiler/blob/master/quras\\_msil/MSIL/Conv\\_Multi.cs#L508](https://github.com/quras-official/quras-smartcontract-compiler/blob/master/quras_msil/MSIL/Conv_Multi.cs#L508))

```
_Convert1by1(Quras.VM.Opcode.INVERT, src, to);
```

```
_Insert1(Quras.VM.Opcode.EQUAL, "", to);
```

Should be changed to :

```
_Convert1by1(Quras.VM.Opcode.EQUAL, src, to);
```

```
_Convert1by1(Quras.VM.Opcode.NOT, null, to);
```

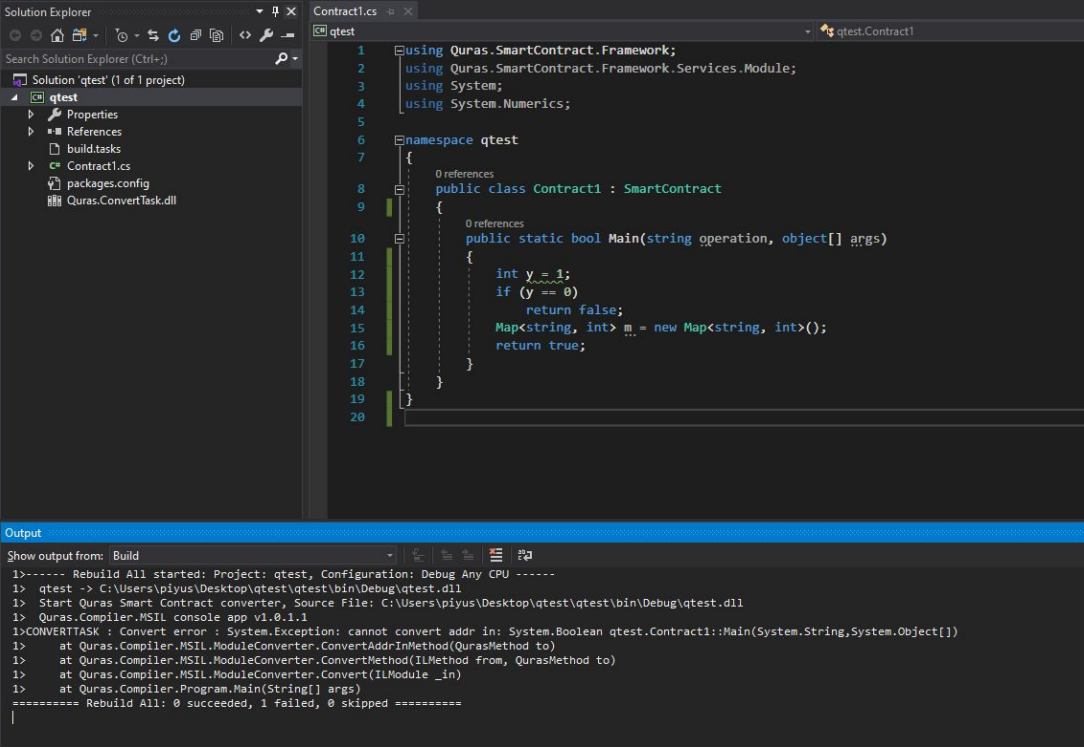
```

496         }
497         else if (src.tokenMethod.Contains("::op_Inequality"))
498         {
499             var _ref = src.tokenUnknown as Mono.Cecil.MethodReference;
500             if (_ref.DeclaringType.FullName == "System.Boolean"
501                 || _ref.DeclaringType.FullName == "System.Int32"
502                 || _ref.DeclaringType.FullName == "System.Numerics.BigInteger")
503             {
504                 _Convert1by1(Quras.VM.Opcode.NUMNOTEQUAL, src, to);
505             }
506             else
507             {
508                 _Convert1by1(Quras.VM.Opcode.INVERT, src, to);
509                 _Insert1(Quras.VM.Opcode.EQUAL, "", to);
510             }

```

## 2) Compilation fails for valid code

Medium



The screenshot shows the Visual Studio IDE with a project named 'qtest'. The code in 'Contract1.cs' is as follows:

```

1 using Quras.SmartContract.Framework;
2 using Quras.SmartContract.Framework.Services.Module;
3 using System;
4 using System.Numerics;
5
6 namespace qtest
7 {
8     0 references
9     public class Contract1 : SmartContract
10     {
11         0 references
12         public static bool Main(string operation, object[] args)
13         {
14             int y = 1;
15             if (y == 0)
16                 return false;
17             Map<string, int> m = new Map<string, int>();
18             return true;
19         }
20     }

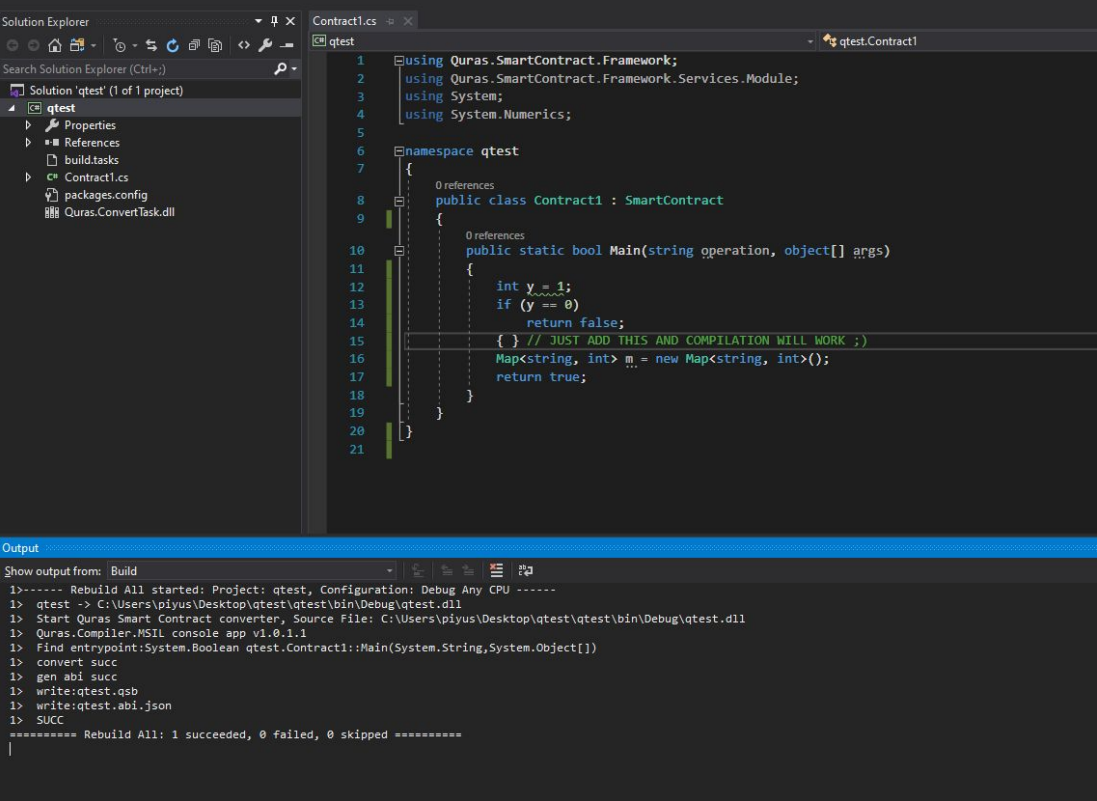
```

The Output window shows the following build output:

```

1>----- Rebuild All started: Project: qtest, Configuration: Debug Any CPU -----
1> qtest -> C:\Users\piyus\Desktop\qtest\qtest\bin\Debug\qtest.dll
1> Start Quras Smart Contract converter, Source File: C:\Users\piyus\Desktop\qtest\qtest\bin\Debug\qtest.dll
1> Quras.Compiler.MSIL console app v1.0.1.1
1> CONVERTTASK : Convert error : System.Exception: cannot convert addr in: System.Boolean qtest.Contract1::Main(System.String,System.Object[])
1> at Quras.Compiler.MSIL.ModuleConverter.ConvertAddrInMethod(QurasMethod to)
1> at Quras.Compiler.MSIL.ModuleConverter.ConvertMethod(ILMethod from, QurasMethod to)
1> at Quras.Compiler.MSIL.ModuleConverter.Convert(ILModule _in)
1> at Quras.Compiler.Program.Main(String[] args)
1> ===== Rebuild All: 0 succeeded, 1 failed, 0 skipped =====

```



The screenshot shows the same Visual Studio IDE with the same project 'qtest'. The code in 'Contract1.cs' is identical to the previous screenshot, but with an additional line of code added in the 'Main' method:

```

1 using Quras.SmartContract.Framework;
2 using Quras.SmartContract.Framework.Services.Module;
3 using System;
4 using System.Numerics;
5
6 namespace qtest
7 {
8     0 references
9     public class Contract1 : SmartContract
10     {
11         0 references
12         public static bool Main(string operation, object[] args)
13         {
14             int y = 1;
15             if (y == 0)
16                 return false;
17             // JUST ADD THIS AND COMPILATION WILL WORK ;)
18             Map<string, int> m = new Map<string, int>();
19             return true;
20         }
21     }

```

The Output window shows the following build output:

```

1>----- Rebuild All started: Project: qtest, Configuration: Debug Any CPU -----
1> qtest -> C:\Users\piyus\Desktop\qtest\qtest\bin\Debug\qtest.dll
1> Start Quras Smart Contract converter, Source File: C:\Users\piyus\Desktop\qtest\qtest\bin\Debug\qtest.dll
1> Quras.Compiler.MSIL console app v1.0.1.1
1> Find entrypoint: System.Boolean qtest.Contract1::Main(System.String,System.Object[])
1> convert succ
1> gen abi succ
1> write:qtest.qsb
1> write:qtest.abi.json
1> SUCC
1> ===== Rebuild All: 1 succeeded, 0 failed, 0 skipped =====

```

Reason :

[https://github.com/quras-official/quras-smartcontract-compiler/blob/master/quras\\_msil/MSIL/Conv\\_Multi.cs#L1081](https://github.com/quras-official/quras-smartcontract-compiler/blob/master/quras_msil/MSIL/Conv_Multi.cs#L1081)

## 3) Variables in Contract creates issue

Low

```
namespace qtest
{
    1 reference
    public class Contract1 : SmartContract
    {
        public static string randomstring = "thisis atest111!!";
        0 references
        public static void Main(string operation, params object[] args)
        {
            Contract2.TestString();
        }
    }

    1 reference
    public class Contract2 : SmartContract
    {
        1 reference
        public static void TestString()
        {
            Runtime.Log(Contract1.randomstring);
        }
    }
}
```

Variables that are not `readonly` **fail silently** without throwing any exception leading to unexpected output.

#### 4) Incorrect SETITEM

Medium

```

namespace qtest
{
    0 references
    public class Contract1 : SmartContract
    {
        0 references
        public static byte[] Main()
        {
            byte[] b = new byte[] { 0x02, 0x02, 0x02, 0x02, 0x02 };
            b[2] = 0x05;
            return b;
        }
    }
}

```

The above Smart Contract leads to following OPCODES

```

1 52
2
3 c5 // new array
4 6b // TOALTSTACK
5 61 // nop
6
7 05
8 02 // DATA
9 02 // DATA
10 02 // DATA
11 02 // DATA
12 02 // DATA
13 6c // FROMALTSTACK
14 76 // DUP
15 6b // TOALTSTACK
16
17 00 // push0
18 52 // push2
19 7a // ROLL
20 c4 // SETITEM
21 6c // FROMALTSTACK
22 76 // dup
23 6b // TOALTSTACK
24 00 // push0
25 c3 // pickitem
26
27 52 // push2
28 55 // push5
29 c4 // setitem
30 6c // FROMALTSTACK
31

```

Here in line 29 SETITEM fails as it is allowed for arrays or maps (not bytearray).

## 5) NULL : The checks doesn't work as expected

Medium

```
namespace qtest
{
    0 references
    public class Contract1 : SmartContract
    {
        1 reference
        public static bool IsNull(byte[] value)
        {
            return value == null;
        }

        0 references
        public static bool Main()
        {
            byte[] b = new byte[] { 0x02, 0x02, 0x02, 0x02, 0x02 };
            b[2] = 0x05;
            return IsNull(b);
        }
    }
}
```

The above code gives following ILCode

The screenshot displays the Visual Studio IDE with the IL code generated from the provided C# code. The main window shows the IL for the `Main` method, and a smaller window shows the IL for the `IsNull` method.

**IL Code for `Contract1.Main`:**

```
.method public hidebysig static bool Main() cil managed
{
    // Code size 34 (0x22)
    .maxstack 3
    .locals init ([0] uint8[] b,
                 [1] bool V_1)
    IL_0000: nop
    IL_0001: ldc.i4.5
    IL_0002: newarr [mscorlib]System.Byte
    IL_0003: dup
    IL_0004: ldtoken field valueType '<PrivateImplementationDetails>'.StaticArrayInitTypeSize=5
    IL_0005: call void [mscorlib]System.Runtime.CompilerServices.RuntimeHelpers::InitializeArray
    IL_0006: stloc.0
    IL_0007: ldloc.0
    IL_0008: ldc.i4.2
    IL_0009: ldc.i4.5
    IL_000A: stelem.i1
    IL_000B: ldloc.0
    IL_000C: call bool qtest.Contract1::IsNull(uint8[])
    IL_000D: stloc.1
    IL_000E: br.s IL_0020
    IL_000F: ldloc.1
    IL_0010: ret
    IL_0020: ret
} // end of method Contract1::Main
```

**IL Code for `Contract1.IsNull`:**

```
.method public hidebysig static bool IsNull(uint8[] 'value') cil managed
{
    // Code size 10 (0x0a)
    .maxstack 2
    .locals init ([0] bool V_0)
    IL_0000: nop
    IL_0001: ldarg.0
    IL_0002: ldnull
    IL_0003: ceq
    IL_0004: stloc.0
    IL_0005: stloc.0
    IL_0006: br.s IL_0008
    IL_0007: ldloc.0
    IL_0008: ret
} // end of method Contract1::IsNull
```



But ceq is defined by NUMEQUAL

```

749         case CodeEx.Ceq:
750             _Convert1by1(Quras.VM.OpCode.NUMEQUAL, src, to);
751             break;
752

```

VM needs to accept 'null' too in NUMEQUAL.

- 6) Issue : The contract should get destroyed after the migration is complete, but the function just returns a boolean value. Here return must be replaced by :  
 Contract\_Destroy(engine);

**NOTE**

<https://github.com/quras-official/quras-blockchain-csharp/blob/master/QurasCore/SmartContract/StateMachine.cs>

```

ContractState contract = contracts.TryGet(hash);
if (contract == null)
{
    contract = new ContractState
    {
        Script = script,
        ParameterList = parameter_list,
        ReturnType = return_type,
        HasStorage = need_storage,
        Name = name,
        CodeVersion = version,
        Author = author,
        Email = email,
        Description = description
    };
    contracts.Add(hash, contract);
    contracts_created.Add(hash, new UInt160(engine.CurrentContext.ScriptHash));
    if (need_storage)
    {
        foreach (var pair in storages.Find(engine.CurrentContext.ScriptHash).ToArray())
        {
            storages.Add(new StorageKey
            {
                ScriptHash = hash,
                Key = pair.Key.Key
            }, new StorageItem
            {
                Value = pair.Value.Value
            });
        }
    }
    engine.EvaluationStack.Push(StackItem.FromInterface(contract));
    return true;
}

```

## 7) Issue : Resource leakage during persisting block in Level DB

Medium

<https://github.com/quras-official/quras-blockchain-csharp/blob/master/QurasCore/Implementations/Blockchains/LevelDB/LevelDBBlockchain.cs>

```

        case TransactionType.PublishTransaction:
        {
#pragma warning disable CS0612
            PublishTransaction publish_tx = (PublishTransaction)tx;
            contracts.GetOrAdd(publish_tx.ScriptHash, () => new ContractState
            {
                Script = publish_tx.Script,
                ParameterList = publish_tx.ParameterList,
                ReturnType = publish_tx.ReturnType,
                HasStorage = publish_tx.NeedStorage,
                Name = publish_tx.Name,
                CodeVersion = publish_tx.CodeVersion,
                Author = publish_tx.Author,
                Email = publish_tx.Email,
                Description = publish_tx.Description
            });
#pragma warning restore CS0612
            break;
        case TransactionType.InvocationTransaction:
        {
            InvocationTransaction itx = (InvocationTransaction)tx;
            CachedScriptTable script_table = new CachedScriptTable(contracts);
            StateMachine service = new StateMachine(accounts, validators, assets, contracts, storages);
            ApplicationEngine engine = new ApplicationEngine(TriggerType.Application, itx, script_table, service, itx.Gas);
            engine.LoadScript(itx.Script, false);
            if (engine.Execute())
            {
                service.Commit();
                notifications.AddRange(service.Notifications);
            }
            break;
        }
    }
}

```

## 8) Issue : Prevent Leakage in application engine during running Smart Contract.

Medium

<https://github.com/quras-official/quras-blockchain-csharp/blob/master/QurasCore/SmartContract/ApplicationEngine.cs>

The StateMachine service should be created by “using” statement for the “new” construct. In order to prevent resource leakage for the interop service.

```

public static ApplicationEngine Run(byte[] script, IScriptContainer container = null)
{
    DataCache<UInt160, AccountState> accounts = Blockchain.Default.CreateCache<UInt160, AccountState>();
    DataCache<ECPoint, ValidatorState> validators = Blockchain.Default.CreateCache<ECPoint, ValidatorState>();
    DataCache<UInt256, AssetState> assets = Blockchain.Default.CreateCache<UInt256, AssetState>();
    DataCache<UInt160, ContractState> contracts = Blockchain.Default.CreateCache<UInt160, ContractState>();
    DataCache<StorageKey, StorageItem> storages = Blockchain.Default.CreateCache<StorageKey, StorageItem>();
    CachedScriptTable script_table = new CachedScriptTable(contracts);
    StateMachine service = new StateMachine(accounts, validators, assets, contracts, storages);
    ApplicationEngine engine = new ApplicationEngine(TriggerType.Application, container, script_table, service, Fixed8.Zero, true);
    engine.LoadScript(script, false);
    engine.Execute();
    return engine;
}

```