# ANALYSIS RESULTS

Branch master

**Date of report creation**

December 18, 2024

# Content

# 1. ASSIGN_NO_REFERENCE_TO_THIS

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Minor | High | true | Quality |

| Position: /.build/ucioption.cpp:149 | Undecided |
|---|---|
| Function | _ZN9Stockfish6OptionaSERKNSt7__cxx1112basic_string IcSt11char_traitsIcESaIcEEE |
| Message about error | Method 'operator=' doesn't return reference to 'this' but rather a reference to some other object. |

```
144   }
145
146 // Updates currentValue and triggers on_change() action.
        It's up to
147 // the GUI to check for option's limits, but we could
        receive new value
148 // from the user by console window, so let's check the
        bounds anyway.
149 Option& Option::operator=(const std::string& v) {
150
151           assert(!type.empty());
152
153           if ((type != "button" && type != "string" && v.empty())
154                   || (type == "check" && v != "true" && v != "false")
```

## 2. BUFFER_OVERFLOW.LOOP

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Critical | Unknown | false | Quality |
| GO | Critical | Unknown | false | Quality |
| JAVA | Major | Unknown | false | Quality |

| Position: /.build/syzygy/tbprobe.cpp:770 | Undecided |
|---|---|
| Function | _ZN9Stockfish12_GLOBAL__N_114do_probe_tableINS0_7TBTableILNS0_6TBTypeE1EEEiEET0_RKNS_8PositionEPT_NS_10Tablebases8WDLScoreEPNSB_10ProbeStateE |
| Message about error | Due to the loop condition dependence on the values of array elements 'd->pieces[i]', the loop at tbprobe.cpp:770 in function 'do_probe_table' can have an unbounded number of iterations, eventually leading to a buffer overflow. (CWE121, CWE122) |

```
765
766    assert(type_of(pc) == PAWN);
767
768 leadPawns = b = pos.pieces(color_of(pc), PAWN); 769 do
```

```
770            squares[size++] = pop_lsb(b) ^ flipSquares;
```

```
771    while (b);
772
773    leadPawnsCnt = size;
774
775 std::swap(squares[0], *std::max_element(squares, squares +
          leadPawnsCnt, pawns_comp));
```

| Position: /.build/syzygy/tbprobe.cpp:918 | Undecided |
|---|---|
| Function | _ZN9Stockfish12_GLOBAL__N_114do_probe_tableINS0_7TBTableILNS0_6TBTypeE1EEEiEET0_RKNS_8PositionEPT_NS_10Tablebases8WDLScoreEPNSB_10ProbeStateE |
| Message about error | Due to the loop condition dependence on the values of array elements 'd->groupLen[next + 1]', the loop at tbprobe.cpp:918 in function 'do_probe_table' can have an unbounded number of iterations, eventually leading to a buffer overflow. (CWE121, CWE122) |

```
913 Square* groupSq = squares + d->groupLen[0];
```

```
914
915 // Encode remaining pawns and then pieces according to
        square, in ascending order
916 bool remainingPawns = entry->hasPawns &&
        entry->pawnCount[1];
917
918 while (d->groupLen[++next])
919    {
920           std::stable_sort(groupSq, groupSq + d->groupLen[next]); uint64_t n = 0;
921
922
923           // Map down a square if "comes later" than a square in the previous
```

## 3. DIVISION_BY_ZERO

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CSHARP | Major | High | true | Quality |
| CXX | Major | Low | true | Quality |
| GO | Major | Low | true | Quality |
| JAVA | Major | Low | true | Quality |
| SCALA | Major | Low | true | Quality |

| Position: /.build/uci.cpp:451 | Undecided |
|---|---|
| Function | void Stockfish::UCIEngine::benchmark(std::istream &) |
| Message about error | Division by zero (CWE369) |

```
446 << "\nThread count 447 <<                    : " << setup.threads : " <<
"\nThread binding 448 << "\nTT                   threadBinding : " <<
size [MiB]                                       setup.ttSize : "
449 << "\nHash max, avg [per mille] 450 << "\n
single search                                    : " << maxHashfull[0] << ",
        "
451 << totalHashfull[0] / numHashfullReadings
452 << "\n            single game                : " << maxHashfull[1] << ",
        "
453 << totalHashfull[1] / numHashfullReadings 454 <<
"\nTotal nodes searched                         : " << nodes
455 << "\nTotal search time [s] 456 <<           : " << totalTime / 1000.0 : " <<
"\nNodes/second                                 1000 * nodes /
        totalTime << std::endl;
```

| Position: /.build/search.cpp:1891 | Undecided |
|---|---|
| Function | Move Stockfish::Search::Skill::pick_best(const RootMoves &, size_t) |
| Message about error | Division by zero (CWE369) |

```
1886 // random. Then we choose the move with the resulting
          highest score.
1887 for (size_t i = 0; i < multiPV; ++i) 1888
          {
1889            // This is our magic formula
```

```
1890        int push = (weakness * int(topScore -
            rootMoves[i].score)
1891                                + delta * (rng.rand<unsigned>() %
                                    int(weakness)))
1892                        / 128;
1893
1894        if (rootMoves[i].score + push >= maxScore) {
1895
1896            maxScore = rootMoves[i].score + push;
```

## 4.DIVISION_BY_ZERO.EX.FLOAT

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Minor | High | true | Quality |
| GO | Minor | High | true | Quality |
| JAVA | Minor | High | true | Quality |
| KOTLIN | Minor | High | true | Quality |
| PYTHON | Minor | High | true | Quality |

| Position: /.build/benchmark.cpp:492 | Undecided |
|---|---|
| Function | _ZN9Stockfish9Benchmark15setup_benchmarkERSi |
| Message about error | Variable totalTime with a floating-point type whose possible value set allows a zero value at benchmark.cpp:479 is used as a denominator at benchmark.cpp:492. The result of division is further used without a check for NaN, which leads to undefined program behavior. |

```
487                 totalTime += correctedTime; ply +=
488                 1;
489             }
490     }
491
492 float timeScaleFactor = static_cast<float>(desiredTimeS *
        1000) / totalTime;
493
494 for (const auto& game : BenchmarkPositions) 495
        {
496             setup.commands.emplace_back("ucinewgame"); int ply
497             = 1;
```

| Position: /.build/numa.h:770 | Undecided |
|---|---|
| Function | _ZNK9Stockfish10NumaConfig24suggests_binding_threadsEm |
| Message about error | Variable *is_node_small.largestNodeSize with floating-point type whose possible value set allows a zero value at numa.h:757 is passed to function at numa.h:770, where it is used as a denominator at numa.h:764. The result of division is further used without a check for NaN, which leads to undefined program behavior. |

```
765                     <= SmallNodeThreshold;
```

```
766         };
767
768         size_t numNotSmallNodes = 0; for
769         (auto&& cpus : nodes)
770                 if (!is_node_small(cpus))
771                         numNotSmallNodes += 1;
772
773         return (numThreads > largestNodeSize / 2 || numThreads
            > = numNotSmallNodes * 4) &&
774                 nodes.size() > 1;
775   }
```

## 5. ENUM_TO_BOOLEAN

*Check that enum expression is used as condition*

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Minor | Unknown | true | Quality |

| Position: /.build/position.cpp:308 | Undecided |
|---|---|
| Function | void Stockfish::Position::set_castling_right(Color, Square) |
| Message about error | cr has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
303         st->castlingRights |= cr;
304         castlingRightsMask[kfrom] |= cr;
305         castlingRightsMask[rfrom] |= cr;
306         castlingRookSquare[cr] = rfrom;
307
308         Square kto = relative_square(c, cr & KING_SIDE ? SQ_G1 : SQ_C1);
309         Square rto = relative_square(c, cr & KING_SIDE ? SQ_F1 : SQ_D1);
310
311         castlingPath[cr] = (between_bb(rfrom, rto) |
            between_bb(kfrom, kto)) & ~(kfrom | rfrom);
312     }
313
```

| Position: /.build/position.cpp:309 | Undecided |
|---|---|
| Function | void Stockfish::Position::set_castling_right(Color, Square) |
| Message about error | cr has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
304         castlingRightsMask[kfrom] |= cr;
305         castlingRightsMask[rfrom] |= cr;
306         castlingRookSquare[cr] = rfrom;
307
308         Square kto = relative_square(c, cr & KING_SIDE ? SQ_G1 : SQ_C1);
309         Square rto = relative_square(c, cr & KING_SIDE ? SQ_F1 : SQ_D1);
310
311         castlingPath[cr] = (between_bb(rfrom, rto) |
            between_bb(kfrom, kto)) & ~(kfrom | rfrom);
312     }
```

```
313
314
```

| Position: /.build/search.cpp:577 | Undecided |
|---|---|
| Function | Value Stockfish::Search::Worker::search(Position &, Stack *, Value, Value, Depth, bool) |
| Message about error | pos.captured_piece() has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
572    ValueList<Move, 32> quietsSearched;
573
574 // Step 1. Initialize node 575 Worker*
thisThread = this; 576 ss->inCheck
                                    = pos.checkers();
577 priorCapture                    = pos.captured_piece();
578    Color us                     = pos.side_to_move();
579    ss->moveCount                = 0;
580    bestValue                    = -VALUE_INFINITE;
581    maxValue                     =VALUE_INFINITE;
582
```

| Position: /.build/search.cpp:639 | Undecided |
|---|---|
| Function | Value Stockfish::Search::Worker::search(Position &, Stack *, Value, Value, Depth, bool) |
| Message about error | ttData.bound has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
634 // to save indentation, we list the condition in all code
        between here and there.
635
636 // At non-PV nodes we check for an early TT cutoff 637 if (!PvNode && !
excludedMove && ttData.depth > depth -
        (ttData.value <= beta)
638             && is_valid(ttData.value) // Can happen when !ttHit or when access race in
                probe()
639             && (ttData.bound & (ttData.value >= beta ? BOUND_LOWER :
                BOUND_UPPER))
640             && (cutNode == (ttData.value >= beta) || depth > 8))
641    {
642             // If ttMove is quiet, update move sorting heuristics on TT hit (~2 Elo)
```

```
643         if (ttData.move && ttData.value >= beta) {
644
```

| Position: /.build/position.cpp:736 | Undecided |
|---|---|
| Function | void Stockfish::Position::do_move(Move, StateInfo &, bool) |
| Message about error | captured has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
731         st->majorPieceKey ^= Zobrist::psq[captured][rfrom] ^
            Zobrist::psq[captured][rto];
732         st->nonPawnKey[us] ^= Zobrist::psq[captured][rfrom] ^
            Zobrist::psq[captured][rto];
733         captured = NO_PIECE;
734     }
735
736 if (captured)
737     {
738         Square capsq = to;
739
740         // If the captured piece is a pawn, update pawn hash key, otherwise

741         // update non-pawn material.
```

| Position: /.build/syzygy/tbprobe.cpp:744 | Undecided |
|---|---|
| Function | Stockfish::Tablebases::WDLScore Stockfish::<anonymous namespace>::do_probe_table(const Position &, Stockfish::<anonymous namespace>::TBTable<Stockfish::<anonymous namespace>::WDL> *, WDLScore, ProbeState *) |
| Message about error | pos.side_to_move() has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
739
740 // A given TB entry like KRK has associated two materials
        keys: KRvk and Kvkr.
741 // If both sides have the same pieces keys are equal. In
        this case TB tables
742 // only stores the 'white to move' case, so if the position
        to look up is black
```

```
743 // to move, we need to switch the color and flip the squares
        before lookup.
744 bool symmetricBlackToMove = (entry->key == entry->key2 &&
        pos.side_to_move());
745
746 // TB files are calculated for white as the stronger side.
        For instance, we
747 // have KRvK, not KvKR. A position where the stronger side
        is white will have
748 // its material key == entry->key, otherwise we have to
        switch the color and
749 // flip the squares before to lookup.
```

| Position: /.build/search.cpp:746 | Undecided |
|---|---|
| Function | Value Stockfish::Search::Worker::search(Position &, Stack *, Value, Value, Depth, bool) |
| Message about error | ttData.bound has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
741         ss->staticEval = eval =
742             to_corrected_static_eval(unadjustedStaticEval,
                * thisThread, pos, ss);
743
744         // ttValue can be used as a better position evaluation (~7 Elo)

745         if (is_valid(ttData.value)
746                 && (ttData.bound & (ttData.value > eval ?
                    BOUND_LOWER : BOUND_UPPER)))
747                 eval = ttData.value;
748     }
749     else
750     {
751         unadjustedStaticEval =
```

| Position: /.build/search.cpp:928 | Undecided |
|---|---|
| Function | Value Stockfish::Search::Worker::search(Position &, Stack *, Value, Value, Depth, bool) |
| Message about error | ttData.bound has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
923
924 moves_loop: // When in check, search starts here
```

```
925
926         // Step 12. A small Probcut idea (~4 Elo) probCutBeta =
927         beta + 417;
928         if ((ttData.bound & BOUND_LOWER) && ttData.depth >= depth - 4
            && ttData.value >= probCutBeta
929                 && !is_decisive(beta) && is_valid(ttData.value) && !
                    is_decisive(ttData.value))
930                 return probCutBeta;
931
932         const PieceToHistory* contHist[] = {(ss - 1)-
            >continuationHistory,
933                                                                 (ss -
                                                                    2)->continuationHist
                                                                    ory,
```

| Position: /.build/position.cpp:954 | Undecided |
|---|---|
| Function | void Stockfish::Position::undo_move(Move) |
| Message about error | st->capturedPiece has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
949   }
950   else
951   {
952         move_piece(to, from);           // Put the piece back to the
            source square
953
954         if (st->capturedPiece)
955         {
956               Square capsq = to;
957
958               if (m.type_of() == EN_PASSANT) {
959
```

| Position: /.build/position.cpp:1074 | Undecided |
|---|---|
| Function | Key Stockfish::Position::key_after(Move) |
| Message about error | captured has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
1069  Square to       = m.to_sq();
1070  Piece  pc       = piece_on(from);
1071  Piece  captured = piece_on(to);
```

```
1072 Key          k               = st->key ^ Zobrist::side;
1073
1074 if (captured)
1075          k ^= Zobrist::psq[captured][to];
1076
1077 k ^= Zobrist::psq[pc][to] ^ Zobrist::psq[pc][from]; 1078

1079 return (captured || type_of(pc) == PAWN) ? k:
         adjust_key50<true>(k);
```

| Position: /.build/search.cpp:1076 | Undecided |
|---|---|
| Function | Value Stockfish::Search::Worker::search(Position &, Stack *, Value, Value, Depth, bool) |
| Message about error | ttData.bound has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
1071 // and lower extension margins scale well. 1072

1073 if (!rootNode && move == ttData.move && !excludedMove 1074
             && depth >= 4 - (thisThread->completedDepth > 33) + ss->ttPv

1075         && is_valid(ttData.value) && !is_decisive(ttData.value)
1076         && (ttData.bound & BOUND_LOWER) && ttData.depth >= depth - 3)

1077    {
1078         Value singularBeta = ttData.value - (56 + 79 * (ss->ttPv && !
             PvNode)) * depth / 64;
1079         Depth singularDepth = newDepth / 2;
1080
1081         ss->excludedMove = move;
```

| Position: /.build/position.cpp:1079 | Undecided |
|---|---|
| Function | Key Stockfish::Position::key_after(Move) |
| Message about error | captured has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
1074         if (captured)
1075              k ^= Zobrist::psq[captured][to];
1076
1077         k ^= Zobrist::psq[pc][to] ^ Zobrist::psq[pc][from];
1078
```

```
1079        return (captured || type_of(pc) == PAWN) ? k :
            adjust_key50<true>(k);
1080    }
1081
1082
1083 // Tests if the SEE (Static Exchange Evaluation) 1084 // value of move
is greater or equal to the given
        threshold. We'll use an
```

| Position: /.build/search.cpp:1531 | Undecided |
|---|---|
| Function | Value Stockfish::Search::Worker::qsearch(Position &, Stack *, Value, Value) |
| Message about error | ttData.bound has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
1526 pvHit              = ttHit && ttData.is_pv;
1527
1528 // At non-PV nodes we check for an early TT cutoff 1529 if (!PvNode
&& ttData.depth >= DEPTH_QS
1530        && is_valid(ttData.value) // Can happen when !ttHit or when access race in
           probe()
1531        && (ttData.bound & (ttData.value >= beta ? BOUND_LOWER :
           BOUND_UPPER)))
1532        return ttData.value;
1533
1534 // Step 4. Static evaluation of the position 1535 Value
unadjustedStaticEval = VALUE_NONE; 1536 if (ss->inCheck)
```

| Position: /.build/search.cpp:1552 | Undecided |
|---|---|
| Function | Value Stockfish::Search::Worker::qsearch(Position &, Stack *, Value, Value) |
| Message about error | ttData.bound has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
1547        ss->staticEval = bestValue =
1548            to_corrected_static_eval(unadjustedStaticEval,
                * thisThread, pos, ss);
1549
1550        // ttValue can be used as a better position evaluation (~13 Elo)
```

```
1551            if (is_valid(ttData.value) && !
                is_decisive(ttData.value)
1552                && (ttData.bound & (ttData.value > bestValue ?
                    BOUND_LOWER : BOUND_UPPER)))
1553                bestValue = ttData.value;
1554    }
1555    else
1556    {
1557            // In case of null move search, use previous static eval with opposite
                sign
```

| Position: /.build/search.cpp:1826 | Undecided |
|---|---|
| Function | void Stockfish::<anonymous namespace>::update_all_stats(const Position &, Stack *, Search::Worker &, Move, Square, ValueList<Move, 32> &, ValueList<Move, 32> &, Depth) |
| Message about error | pos.captured_piece() has enum type which is not isomorphic with boolean type, however it is used as a boolean expression |

```
1821            captureHistory[moved_piece][bestMove.to_sq()][captured] << bonus;

1822    }
1823
1824 // Extra penalty for a quiet early move that was not a TT
            move in
1825 // previous ply when it gets refuted.
1826 if (prevSq != SQ_NONE && ((ss - 1)->moveCount == 1 + (ss -
            1)->ttHit) && !pos.captured_piece())
1827            update_continuation_histories(ss - 1,
                pos.piece_on(prevSq), prevSq, -malus);
1828
1829 // Decrease stats for all non-best capture moves 1830 for (Move
move : capturesSearched)
1831 {
```

## 6. INTEGER_OVERFLOW

*A possible integral overflow in additions or multiplications.*

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CSHARP | Normal | Unknown | true | Quality |
| CXX | Major | Unknown | true | Quality |

| Position: /.build/bitboard.h:368 | Undecided |
|---|---|
| Function | Square Stockfish::pop_lsb(Bitboard &) |
| Message about error | An integer underflow may occur due to arithmetic operation (unsigned subtraction) between variable 'b' and value '1', when 'b' is in range { [1, 18446744073709551615] } (CWE125, CWE190, CWE191) |

```
363
364 // Finds and clears the least significant bit in a non-zero
        bitboard.
365 inline Square pop_lsb(Bitboard& b) { 366
                assert(b);
367             const Square s = lsb(b);
368             b &= b - 1;
369             return s;
370     }
371
372     }     // namespace Stockfish
373
```

| Position: /.build/syzygy/tbprobe.cpp:887 | Undecided |
|---|---|
| Function | int Stockfish::<anonymous namespace>::do_probe_table(const Position &, Stockfish::<anonymous namespace>::TBTable<Stockfish::<anonymous namespace>::DTZ> *, WDLScore, ProbeState *) |
| Message about error | Undefined behavior: an integer underflow may occur due to arithmetic operation (multiplication) between values  and '62', where the first value comes from the expression 'MapA1D1D4[squares[0]] * 63 + (squares[1] - adjust1)' (CWE125, CWE190, CWE191) |

```
882
883 // First piece is below a1-h8 diagonal. MapA1D1D4[] maps the
        b1-d1-d3
```

```
884 // triangle to 0...5. There are 63 squares for second piece
        and 62
885 // (mapped to 0...61) for the third. 886 if
(off_A1H8(squares[0]))
887             idx = (MapA1D1D4[squares[0]] * 63 + (squares[1] - adjust1)) * 62 +
                squares[2] - adjust2;
888
889 // First piece is on a1-h8 diagonal, second below: map this
        occurrence to
890 // 6 to differentiate from the above case, rank_of() maps
        a1-d4 diagonal
891 // to 0...3 and finally MapB1H1H7[] maps the b1-h1-h7
        triangle to 0..27.
892     else if (off_A1H8(squares[1]))
```

| Position: /.build/syzygy/tbprobe.cpp:899 | Undecided |
|---|---|
| Function | int Stockfish::<anonymous namespace>::do_probe_table(const Position &, Stockfish::<anonymous namespace>::TBTable<Stockfish::<anonymous namespace>::DTZ> *, WDLScore, ProbeState *) |
| Message about error | Undefined behavior: an integer underflow may occur due to arithmetic operation (multiplication) between values and '28', where the first value comes from the expression 'rank_of(squares[1]) - adjust1' (CWE125, CWE190, CWE191) |

```
894                 - adjust2;
895
896 // First two pieces are on a1-h8 diagonal, third below 897
        else if (off_A1H8(squares[2]))
898             idx = 6 * 63 * 62 + 4 * 28 * 62 + rank_of(squares[0]) * 7 * 28
899                 + (rank_of(squares[1]) - adjust1) * 28 +
                    MapB1H1H7[squares[2]];
900
901 // All 3 pieces on the diagonal a1-h8 902
        else
903             idx = 6 * 63 * 62 + 4 * 28 * 62 + 4 * 7 * 28 +
                rank_of(squares[0]) * 7 * 6
904                 + (rank_of(squares[1]) - adjust1) * 6 +
                    (rank_of(squares[2]) - adjust2);
```

| Position: /.build/syzygy/tbprobe.cpp:904 | Undecided |
|---|---|

| Function | int Stockfish::<anonymous namespace>::do_probe_table(const Position &, Stockfish::<anonymous namespace>::TBTable<Stockfish::<anonymous namespace>::DTZ> *, WDLScore, ProbeState *) |
|---|---|
| Message about error | Undefined behavior: an integer underflow may occur due to arithmetic operation (multiplication) between values and '6', where the first value comes from the expression 'rank_of(squares[1]) - adjust1' (CWE125, CWE190, CWE191) |

```
899                     + (rank_of(squares[1]) - adjust1) * 28 +
                        MapB1H1H7[squares[2]];
900
901         // All 3 pieces on the diagonal a1-h8 else
902
903             idx = 6 * 63 * 62 + 4 * 28 * 62 + 4 * 7 * 28 +
                    rank_of(squares[0]) * 7 * 6
904                     + (rank_of(squares[1]) - adjust1) * 6 +
                        (rank_of(squares[2]) - adjust2);
905   }
906   else
907         // We don't have at least 3 unique pieces, like in KRRvKBB, just map

908         // the kings.
909         idx = MapKK[MapA1D1D4[squares[0]]][squares[1]];
```

| Position: /.build/syzygy/tbprobe.cpp:1650 | Undecided |
|---|---|
| Function | bool Stockfish::Tablebases::root_probe(Position &, Search::RootMoves &, bool, bool) |
| Message about error | Undefined behavior: an integer underflow may occur due to arithmetic operation (multiplication) between values { [-2147483648, 2147483647] } and '2', where the first value comes from the expression '-dtz' (CWE125, CWE190, CWE191) |

```
1645
1646 // Better moves are ranked higher. Certain wins are ranked
         equally.
1647 // Losing moves are ranked equally unless a 50-move draw is
         in sight.
1648 int r              = dtz > 0 ? (dtz + cnt50 <= 99 && !rep ? MAX_DTZ -
         (rankDTZ ? dtz : 0)
1649                                                              : MAX_DTZ /
                                                                  2 - (dtz +
                                                                  cnt50))
```

```
1650                    : dtz < 0 ? (-dtz * 2 + cnt50 < 100 ? -MAX_DTZ - (rankDTZ ? dtz : 0)
1651                                                                                  : -MAX_DTZ / 2
                                                                                      + (-dtz +
                                                                                      cnt50))
1652                                    : 0;
1653    m.tbRank = r;
1654
1655 // Determine the score to be displayed for this move.
          Assign at least
```

## 7. LOCAL_VAR.HUGE

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Minor | Average | true | Quality |

| Position: /.build/bitboard.cpp:154 | Undecided |
|---|---|
| Function | _ZN9Stockfish12_GLOBAL__N_111init_magicsENS_9PieceTypeEPmPA2_NS_5MagicE |
| Message about error | The size of variable 'reference' is 32768 bytes, which is too big (defined at bitboard.cpp:154). |

```
149                                                 {728, 10316, 55013, 32803, 12281,
                                                    15100, 16645, 255}};
150
151         Bitboard occupancy[4096]; int
152                         epoch[4096] = {}, cnt = 0;
153   #endif
154         Bitboard reference[4096];
155         int             size = 0;
156
157         for (Square s = SQ_A1; s <= SQ_H8; ++s) {
158
159               // Board edges are not considered in the relevant occupancies
```

# 8. NO_CAST.INTEGER_OVERFLOW

*Find situations where the value of an arithmetic expression might overflow before the result is widened to a larger data type*

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CSHARP | Normal | Average | true | Quality |
| CXX | Major | Unknown | true | Quality |

| Position: /.build/syzygy/tbprobe.cpp:887 | Undecided |
|---|---|
| Function | Stockfish::Tablebases::WDLScore Stockfish::<anonymous namespace>::do_probe_table(const Position &, Stockfish::<anonymous namespace>::TBTable<Stockfish::<anonymous namespace>::WDL> *, WDLScore, ProbeState *) |
| Message about error | The value of an arithmetic expression '(MapA1D1D4[squares[0]] * 63 + (squares[1] - adjust1)) * 62 + squares[2] - adjust2' is a subject to overflow because its operands are not cast to a larger data type before performing arithmetic (CWE190, CWE197) |

```
882
883 // First piece is below a1-h8 diagonal. MapA1D1D4[] maps the
        b1-d1-d3
884 // triangle to 0...5. There are 63 squares for second piece
        and 62
885 // (mapped to 0...61) for the third. 886 if
(off_A1H8(squares[0]))
887         idx = (MapA1D1D4[squares[0]] * 63 + (squares[1] - adjust1)) * 62 +
            squares[2] - adjust2;
888
889 // First piece is on a1-h8 diagonal, second below: map this
        occurrence to
890 // 6 to differentiate from the above case, rank_of() maps
        a1-d4 diagonal
891 // to 0...3 and finally MapB1H1H7[] maps the b1-h1-h7
        triangle to 0..27.
892     else if (off_A1H8(squares[1]))
```

| Position: /.build/syzygy/tbprobe.cpp:893 | Undecided |
|---|---|

| Function | Stockfish::Tablebases::WDLScore Stockfish::<anonymous namespace>::do_probe_table(const Position &, Stockfish::<anonymous namespace>::TBTable<Stockfish::<anonymous namespace>::WDL> *, WDLScore, ProbeState *) |
|---|---|
| Message about error | The value of an arithmetic expression '(6 * 63 + rank_of(squares[0]) * 28 + MapB1H1H7[squares[1]]) * 62 + squares[2] - adjust2' is the subject to overflow because its operands are not cast to a larger data type before performing arithmetic (CWE190, CWE197) |

```
888
889 // First piece is on a1-h8 diagonal, second below: map this occurrence to
890 // 6 to differentiate from the above case, rank_of() maps a1-d4 diagonal
891 // to 0...3 and finally MapB1H1H7[] maps the b1-h1-h7 triangle to 0..27.
892     else if (off_A1H8(squares[1]))
893         idx = (6 * 63 + rank_of(squares[0]) * 28 + MapB1H1H7[squares[1]]) * 62 + squares[2]
894                 - adjust2;
895
896 // First two pieces are on a1-h8 diagonal, third below 897
        else if (off_A1H8(squares[2]))
898         idx = 6 * 63 * 62 + 4 * 28 * 62 + rank_of(squares[0]) * 7 * 28
```

| Position: /.build/syzygy/tbprobe.cpp:898 | Undecided |
|---|---|
| Function | Stockfish::Tablebases::WDLScore Stockfish::<anonymous namespace>::do_probe_table(const Position &, Stockfish::<anonymous namespace>::TBTable<Stockfish::<anonymous namespace>::WDL> *, WDLScore, ProbeState *) |
| Message about error | The value of an arithmetic expression '6 * 63 * 62 + 4 * 28 * 62 + rank_of(squares[0]) * 7 * 28 + (rank_of(squares[1]) - adjust1) * 28 + MapB1H1H7[squares[2]]' is a subject to overflow because its operands are not cast to a larger data type before performing arithmetic (CWE190, CWE197) |

```
893         idx = (6 * 63 + rank_of(squares[0]) * 28 + MapB1H1H7[squares[1]]) * 62 + squares[2]
894                 - adjust2;
```

```
895
896 // First two pieces are on a1-h8 diagonal, third below 897 else if
(off_A1H8(squares[2]))
898         idx = 6 * 63 * 62 + 4 * 28 * 62 + rank_of(squares[0]) * 7 * 28
899                 + (rank_of(squares[1]) - adjust1) * 28 +
                    MapB1H1H7[squares[2]];
900
901 // All 3 pieces on the diagonal a1-h8 902
      else
903         idx = 6 * 63 * 62 + 4 * 28 * 62 + 4 * 7 * 28 +
            rank_of(squares[0]) * 7 * 6
```

| Position: /.build/syzygy/tbprobe.cpp:903 | Undecided |
|---|---|
| Function | Stockfish::Tablebases::WDLScore Stockfish::<anonymous namespace>::do_probe_table(const Position &, Stockfish::<anonymous namespace>::TBTable<Stockfish::<anonymous namespace>::WDL> *, WDLScore, ProbeState *) |
| Message about error | The value of an arithmetic expression '6 * 63 * 62 + 4 * 28 * 62 + 4 * 7 * 28 + rank_of(squares[0]) * 7 * 6 + (rank_of(squares[1]) - adjust1) * 6 + (rank_of(squares[2]) - adjust2)' is a subject to overflow because its operands are not cast to a larger data type before performing arithmetic (CWE190, CWE197 ) |

```
898                 idx = 6 * 63 * 62 + 4 * 28 * 62 +
                    rank_of(squares[0]) * 7 * 28
899                         + (rank_of(squares[1]) - adjust1) * 28 +
                            MapB1H1H7[squares[2]];
900
901         // All 3 pieces on the diagonal a1-h8 else
902
903                 idx = 6 * 63 * 62 + 4 * 28 * 62 + 4 * 7 * 28 +
                    rank_of(squares[0]) * 7 * 6
904                         + (rank_of(squares[1]) - adjust1) * 6 +
                            (rank_of(squares[2]) - adjust2);
905     }
906     else
907         // We don't have at least 3 unique pieces, like in KRRvKBB, just map

908         // the kings.
```

## 9. OVERFLOW_UNDER_CHECK

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Critical | Average | true | Quality |
| GO | Critical | Average | true | Quality |
| JAVA | Major | Average | true | Quality |

| Position: /.build/numa.h:411 | Undecided |
|---|---|
| Function | _ZN9Stockfish20get_process_affinityEv |
| Message about error | Accessing an element of array 'mask->__bits' of size 16 at numa.h:411 can lead to a buffer overflow, since the index '__cpu / 64' can have an out of range value 1023, as indicated by a preceding conditional expression at numa.h:410. (CWE119, CWE121, CWE122, CWE124, CWE194, CWE195) |

```
406         CPU_FREE(mask);
407         std::exit(EXIT_FAILURE);
408   }
409
410 for (CpuIndex c = 0; c < MaxNumCpus; ++c)
411         if (CPU_ISSET_S(c, masksize, mask))
412               cpus.insert(c);
413
414   CPU_FREE(mask);
415
416   return cpus;
```

## 10. UNCHECKED_FUNC_RES.LIB.STRICT

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Minor | VeryHigh | true | Quality |
| JAVA | Minor | VeryHigh | true | Quality |
| KOTLIN | Minor | VeryHigh | true | Quality |
| SCALA | Minor | VeryHigh | true | Quality |

| Position: /.build/syzygy/tbprobe.cpp:235 | Undecided |
|---|---|
| Function | _ZN9Stockfish12_GLOBAL__N_16TBFile3mapEPPvPmNS0_6TBTypeE |
| Message about error | Return value of function 'fstat', called at tbprobe.cpp:235, is not checked. The return value possibly contains an error code and ignoring it may lead to missing important errors. (CWE252, CWE754) |

```
230 int                    fd = ::open(fname.c_str(), O_RDONLY);
231
232 if (fd == -1) 233
                return *baseAddress = nullptr, nullptr;
234
235 fstat(fd, &statbuf);
236
237 if (statbuf.st_size % 64 != 16) 238
        {
239             std::cerr << "Corrupt tablebase file " << fname << std::endl;

240             exit(EXIT_FAILURE);
```

## 11. UNCHECKED_FUNC_RES.STAT

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Normal | High | true | Quality |
| JAVA | Normal | High | true | Quality |
| KOTLIN | Normal | High | true | Quality |
| SCALA | Normal | High | true | Quality |

| Position: /.build/position.cpp:929 | Undecided |
|---|---|
| Function | _ZN9Stockfish8Position9undo_moveENS_4MoveE |
| Message about error | Return value of function 'Stockfish::Position::piece_on', called at position.cpp:929, is not checked, but it is usually checked for this function (6/7). (CWE252, CWE754) |

```
924    sideToMove = ~sideToMove;
925
926    Color us            = sideToMove;
927 Square from = m.from_sq(); 928
Square to = m.to_sq();
929 Piece pc               = piece_on(to);
930
931 assert(empty(from) || m.type_of() == CASTLING); 932
        assert(type_of(st->capturedPiece) != KING);
933
934 if (m.type_of() == PROMOTION)
```

## 12. UNINIT.CTOR

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Major | High | true | Quality |

| Position: /.build/ucioption.cpp:76 | Undecided |
|---|---|
| Function | _ZN9Stockfish6OptionC2EPKcSt8functionIFSt8optionalINSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEEERKS0_EE |
| Message about error | Constructor declared at ucioption.cpp:76 may not initialize class members of 'Stockfish::Option'. The following members aren't initialized: idx. (CWE457) |

```
71 std::size_t OptionsMap::count(const std::string& name) const
       { return options_map.count(name); }
72
73 Option::Option(const OptionsMap* map) : 74
               parent(map) {}
75
76 Option::Option(const char* v, OnChange f) :
77             type("string"),
78             min(0),
79             max(0),
80             on_change(std::move(f)) { defaultValue =
81             currentValue = v;
```

| Position: /.build/ucioption.cpp:84 | Undecided |
|---|---|
| Function | _ZN9Stockfish6OptionC2EbSt8functionIFSt8optionalINSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEEERKS0_EE |
| Message about error | Constructor declared at ucioption.cpp:84 may not initialize class members of 'Stockfish::Option'. The following members aren't initialized: idx. (CWE457) |

```
79             max(0),
80             on_change(std::move(f)) { defaultValue =
81             currentValue = v;
82     }
83
84 Option::Option(bool v, OnChange f) :
85             type("check"),
86             min(0),
87             max(0),
88             on_change(std::move(f)) {
```

```
89          defaultValue = currentValue = (v ? "true" : "false");
```

| Position: /.build/search.h:85 | Undecided |
|---|---|
| Function | _ZN9Stockfish6Search8RootMoveC2ENS_4MoveE |
| Message about error | Constructor declared at search.h:85 may not initialize class members of 'Stockfish::Search::RootMove'. The following members aren't initialized: tbScore. (CWE457) |

```
80 // RootMove struct is used for moves at the root of the tree.
       For each root move
81 // we store a score and a PV (really a refutation in the case
       of moves which
82 // fail low). Score is normally set at -VALUE_INFINITE for
       all non-pv moves.
83    struct RootMove {
84
85          explicit RootMove(Move m):
86                  pv(1, m) {}
87          bool extract_ponder_from_tt(const TranspositionTable& tt, Position& pos);

88          bool operator==(const Move& m) const { return pv[0] == m; }

89          // Sort in descending order
90          bool operator<(const RootMove& m) const {
```

| Position: /.build/ucioption.cpp:92 | Undecided |
|---|---|
| Function | _ZN9Stockfish6OptionC2ESt8functionIFSt8optionalINSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEEERKS0_EE |
| Message about error | Constructor declared at ucioption.cpp:92 may not initialize class members of 'Stockfish::Option'. The following members aren't initialized: idx. (CWE457) |

```
87          max(0),
88          on_change(std::move(f)) {
89          defaultValue = currentValue = (v ? "true" : "false");
90    }
91
92 Option::Option(OnChange f) :
93          type("button"),
94          min(0),
95          max(0),
96          on_change(std::move(f)) {}
```

97

| Position: /.build/ucioption.cpp:98 | Undecided |
|---|---|
| Function | _ZN9Stockfish6OptionC2EdiiSt8functionIFSt8optionalINSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEEERKS0_EE |
| Message about error | Constructor declared at ucioption.cpp:98 may not initialize class members of 'Stockfish::Option'. The following members aren't initialized: idx. (CWE457) |

```
 93         type("button"),
 94         min(0),
 95         max(0),
 96         on_change(std::move(f)) {}
 97
 98 Option::Option(double v, int minv, int maxv, OnChange f) :
 99         type("spin"),
100         min(minv),
101         max(maxv),
102         on_change(std::move(f)) {
103         defaultValue = currentValue = std::to_string(v);
```

| Position: /.build/ucioption.cpp:106 | Undecided |
|---|---|
| Function | _ZN9Stockfish6OptionC2EPKcS2_St8functionIFSt8optionalINSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEEERKS0_EE |
| Message about error | Constructor declared at ucioption.cpp:106 may not initialize class members of 'Stockfish::Option'. The following members aren't initialized: idx. (CWE457) |

```
101         max(maxv),
102         on_change(std::move(f)) {
103         defaultValue = currentValue = std::to_string(v);
104     }
105
106 Option::Option(const char* v, const char* cur, OnChange f) :
107         type("combo"),
108         min(0),
109         max(0),
110         on_change(std::move(f))
111         { defaultValue = v;
```

## 13. UNINIT_HEAP.CCTOR

*Memory was allocated in constructor, but don't allocates in copy constructor.*

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Major | Unknown | true | Quality |

| Position: /.build/tt.h:72 | Undecided |
|---|---|
| Function | _ZN9Stockfish18TranspositionTableD2Ev |
| Message about error | Copy constructor doesn't exist for class 'Stockfish::TranspositionTable' which destructor deallocates class-member 'this->table' at tt.h:72. |

```
67
68
69  class TranspositionTable {
70
71        public:
72          ~TranspositionTable() { aligned_large_pages_free(table); }
73
74          void resize(size_t mbSize, ThreadPool& threads); TT size                // Set
75          void clear(ThreadPool& threads); Re-initialize                          //
            memory, multithreaded int hashfull(int
76          maxAge = 0)
77              const; // Approximate what fraction of entries (permille) have been
                written to during this root search
```

## 14. UNREACHABLE_CODE

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CSHARP | Normal | Average | true | Quality |
| CXX | Normal | Average | true | Quality |
| GO | Normal | Average | true | Quality |
| JAVA | Normal | Average | true | Quality |
| KOTLIN | Normal | Average | true | Quality |

| Position: /.build/uci.cpp:441 | Undecided |
|---|---|
| Function | _ZN9Stockfish9UCIEngine9benchmarkERSi |
| Message about error | The condition in ternary operator is always true at uci.cpp:441. (CWE561) |

```
436    std::cerr << "============================"
437                    << "\nVersion                          : "
438                    << engine_version_info() //
439                    "\nCompiled by                      : "
440                    << compiler_info()
441                    << "Large pages                      : " <<
                       (has_large_pages() ? "yes" : "no")
442                    << "\nUser invocation                  : " <<
                       BenchmarkCommand << " "
443                    << setup.originalInvocation << "\nFilled invocation
                                                  : " << BenchmarkCommand
444                    << " " << setup.filledInvocation <<
445                    "\nAvailable processors             : " <<
                       engine.get_numa_config_as_string() <<
446                    "\nThread count                     : " <<
                       setup.threads
```

| Position: /.build/numa.h:538 | Undecided |
|---|---|
| Function | _ZN9Stockfish10NumaConfig11from_systemEb |
| Message about error | This statement in the source code might be unreachable during program execution. (CWE561) |

```
533                }
534            }
535
536            if (useFallback)
```

```
537                 {
538                     for (CpuIndex c = 0; c < SYSTEM_THREADS_NB; ++c)
539                         if (is_cpu_allowed(c))
540                             cfg.add_cpu_to_node(NumaIndex{0}, c);
541                 }
542
543     # elif defined(_WIN64)
```

## 15. UNUSED_FUNC_RES.REWRITE.MINOR

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CXX | Minor | Average | true | Quality |
| JAVA | Minor | Average | true | Quality |
| KOTLIN | Minor | Average | false | Quality |
| SCALA | Minor | Average | true | Quality |

| Position: /.build/position.cpp:929 | Undecided |
|---|---|
| Function | _ZN9Stockfish8Position9undo_moveENS_4MoveE |
| Message about error | Return value of function 'Stockfish::Position::piece_on' passed to 'pc' at position.cpp:929 will be rewritten later. (CWE563) |

```
924    sideToMove = ~sideToMove;
925
926    Color us            = sideToMove;
927 Square from = m.from_sq(); 928
Square to = m.to_sq();
929 Piece pc              = piece_on(to);
930
931 assert(empty(from) || m.type_of() == CASTLING); 932
        assert(type_of(st->capturedPiece) != KING);
933
934 if (m.type_of() == PROMOTION)
```

## 16. UNUSED_VALUE

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CSHARP | Minor | Unknown | true | Quality |
| CXX | Minor | Average | true | Quality |
| JAVA | Minor | Average | true | Quality |

| Position: /.build/search.cpp:858 | Undecided |
|---|---|
| Function | _ZN9Stockfish6Search6Worker6searchILNS_8NodeTypeE2EEEiRNS_8PositionEPNS0_5StackEiiib |
| Message about error | The current value of 'probCutBeta' is not used as it will be overwritten. (CWE325, CWE563) |

```
853         depth -= 1 + !ttData.move;
854
855 // Step 11. ProbCut (~10 Elo)
856 // If we have a good enough capture (or queen promotion) and
        a reduced search
857 // returns a value much above beta, we can (almost) safely
        reverse the previous move.
858 probCutBeta = beta + 187 - 56 * improving;
859 if (!PvNode && depth > 3,860
                && !is_decisive(beta)
861             // If value from transposition table is lower than probCutBeta,
                don't attempt
862             // probCut there and in further interactions with transposition
                table cutoff
863             // depth is set to depth - 3 because probCut search has depth set to depth
                - 4
```

## 17. WRONG_ARGUMENTS_ORDER

*Check for mismatched variables, passed as call arguments*

| Language | Seriousness | Reliability | Status | Situation |
|---|---|---|---|---|
| CSHARP | Normal | Unknown | true | Quality |
| CXX | Major | Unknown | true | Quality |
| JAVA | Major | Unknown | true | Quality |
| KOTLIN | Major | Unknown | true | Quality |
| PYTHON | Major | Unknown | true | Quality |

| Position: /.build/search.cpp:1247 | Undecided |
|---|---|
| Function | Value Stockfish::Search::Worker::search(Position &, Stack *, Value, Value, Depth, bool) |
| Message about error | Possibly mismatched call arguments in function 'search': 'alpha' and 'beta' passed in place of 'Value beta' and 'Value alpha' |

```
1242
1243        // Extend move from transposition table if we are about to dive into
            qsearch.
1244        if (move == ttData.move && ss->ply <=
            thisThread->rootDepth * 2)
1245            newDepth = std::max(newDepth, 1);
1246
1247        value = -search<PV>(pos, ss + 1, -beta, -alpha, newDepth, false);
1248    }
1249
1250 // Step 19. Undo move 1251
        pos.undo_move(move);
1252
```