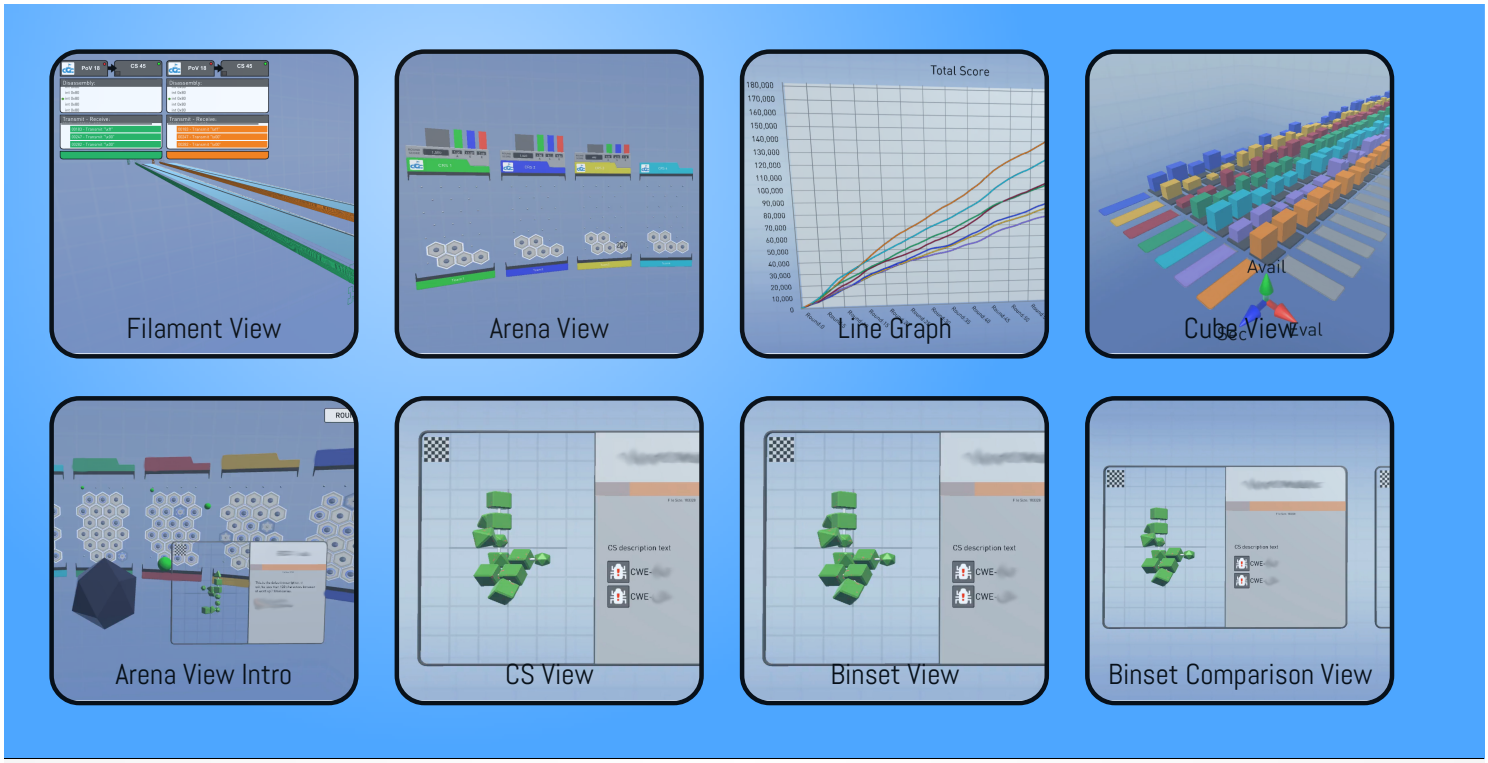
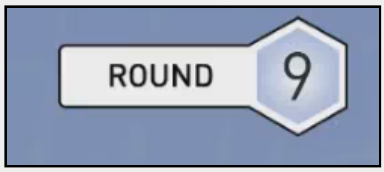


TOP

Global Arguments

Arguments that are handled by the environment, and not a specific package.

- The appropriateness of these varies.
- All argument names are case insensitive.

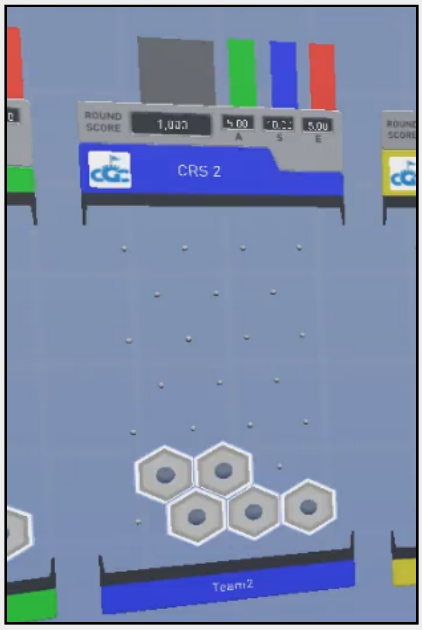


Arguments:

- *-RoundLabel=9*

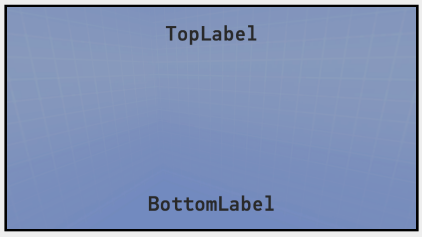
Arguments:

- *-Round=2*
- *-UseRoundAsRoundLabel*



Arguments:

- *-Anonymize*



Arguments:

- *-TopLabel="TopLabel"*
- *-BottomLabel="BottomLabel"*

Optional Arguments

-RoundLabel = `9`

This will cause the round label to appear in the top-right corner, indicating the specified value.

-UseRoundAsRoundLabel

(No arguments) If present, this will cause the round label to appear in the top-right corner, using the value passed into -Round as the text.

This is useful for packages that take a -Round argument, where -RoundLabel would be redundant.

-Anonymize

(No arguments) If present, this will use "Team #" instead of real team names, "CRS #" instead of real CRS names, primary colors instead of team colors, and the CGC logo instead of team logos.

Does not affect Challenge Sets.

If anonymity is critical, proof the resulting video!

-TopLabel = `"Any String"`

A caption that will appear across the top of the screen for the duration of the video.

Surround in quotes.

-BottomLabel = `"Any String"`

A caption that will appear across the bottom of the screen for the duration of the video.

-BottomRightLabel = `"Any String"`

A caption that will appear across the bottom-right of the screen for the duration of the video.

-CuratedOverride = `(-8.4, 4.9, 20.0):(0.5, -0.2, -0.8):2.5:1.5|(-8.4, 4.9, 20.0):(-0.4, -0.3, -0.9):2.5:1.5`

An argument that accepts a serialized set of choreography sub-steps. The presence of this argument overrides the regular choreography, allowing an author to set up and utilize simple choreography sequences designed in the standalone. This argument is not globally supported but implemented on a case-by-case basis. To implement this argument in a new package, use an IfCommandLineExists step followed by a CuratedChoreography step, as demonstrated in FilamentView.json.

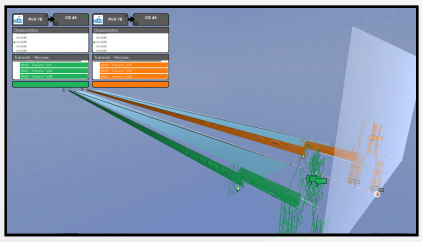
TOP

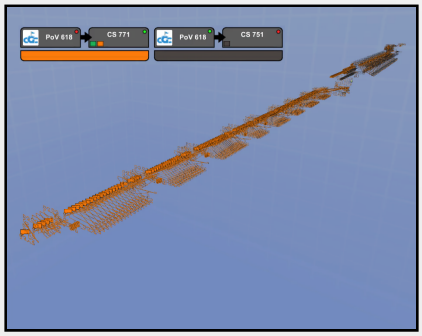
Filament View

Haxxis Package Name: FilamentView/FilamentView.json

A graphical representation of an execution (the interaction between a Challenge Set and a Poll/PoV).

- Can show multiple executions.
- Instruction limit is 300,000, and 100,000 is about as long as can be while still being intelligible.
- Can show Disassembly output from the execution.
- Can show Communications (input and output) between the Poll/PoV and the Challenge Set.
- Can show the Memory View, which maps reads and writes across the filament.
- The cursor (Dot and Plane) indicate the current instruction. The Disassembly and Communications windows display the data around that instruction. A thin line marks the progress of the cursor through the memory space.
- The filament is annotated with icons that represent syscalls and other notable events.
- Does not show dot cursor or plane if neither disasm nor comms are enabled.
- Use with -BottomLabel or -BottomRightLabel so as not to obstruct comms/disasm windows.
- Can be used interactively with the InteractiveCursor argument.
- If the trace can't be pulled from the Trace API, try disabling memory view with -ShowMemory=false .



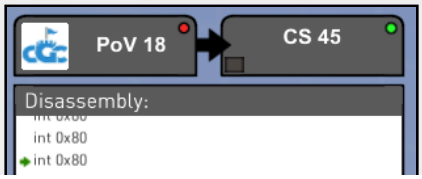


Filament view showing the same PoV against two CSs (a ref and a patch), with one succeeding.

Arguments:

- *-BinaryIds=771,751*
- *-RequestIds=618,618*
- *-RequestNatures=pov,pov*
- *-ShowDisassembly=false*
- *-ShowComms=false*
- *-ShowMemory=false*
- *-OverlayTraces=true*
- *-TraceTeamIds=5*

Note the orange and green boxes under the left CS header. This indicates two teams have fielded this same CS.



Required Arguments

-BinaryIds = 1 3,4
CSV (one to four values). The Binary IDs in question. For multiple executions, you can specify the same value multiple times.

-RequestIds = 5 984,984
CSV (one to four values). The corresponding PoV or Poll IDs in question. For multiple executions, you can specify the same value multiple times.

Optional Arguments

-StartInstruction = 0
The instruction to start the cursor on.

-InstructionCount = 0
How many instructions to step the cursor over (NOT how many steps to take).

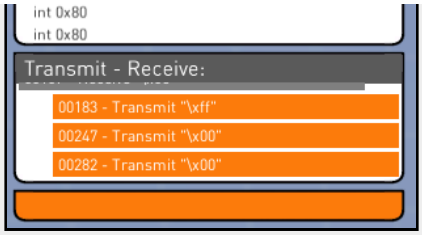
-InstructionsPerStep = 30
The number of instructions to jump over with east cursor step.

-SecondsPerStep = 0.1
How long to take between cursor steps.

-Natures = pov poll poll,poll
CSV (one to four values). The corresponding request natures (poll or pov) of the given requests. For multiple executions, you can specify the same value multiple times.

-TraceTeamIds = 8 2,4 3,3,3
The team IDs to use for coloring the Challenge Set comms, and the filament. 8 is 'Reference' gray.

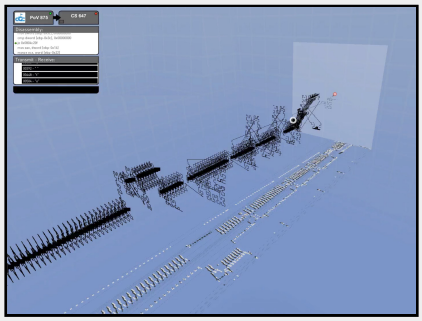
-ExecutionIds = -1 -1,-1,-1 40,-1 5030
CSV (one to four values). The Execution IDs in question.



Disassembly and Communications Windows showing Reference PoV 18 against Reference CS 45.

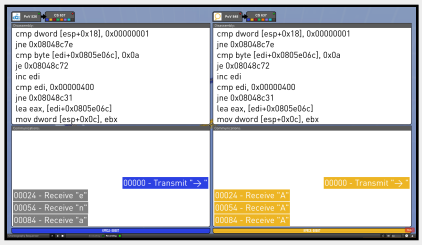
The orange bar across the bottom correlates this window to the orange filament.

The small, colored squares under the CS label indicate the team(s) who have deployed that version of the CS.



The sequence of white and black marks along-side the filament is the Memory View.

The white marks represent reads, and the black marks represent writes. Each of the rectangular planes along the bottom denotes a single memory allocation, with the width indicating the size of the allocation. The thin black lines that run parallel to the filament denote discontinuities in memory space.



Filament view showing the large-form disassembly/comms window.

Supports one or two executions. Any more will likely cause disassembly lines to be truncated.

This will cause the filament to not display, as it would be mostly occluded anyway.

Arguments:

- ***-LargeDisasm=true***
- *-BinaryIds=637,637*
- *-RequestIds=526,848*
- *-RequestNatures=pov,pov*
- *-ShowDisassembly=true*
- *-ShowComms=true*
- *-TraceTeamIds=1,2*

The outcomes of service requests are

Passing in a value other than -1 will fetch the trace directly by execution ID (usually the Execution ID is determined by the Binary and Request IDs).

Binary and Request IDs cannot be inferred from an Execution ID, which will cause default values to appear in the header of the disassembly window.

-InstructionSkips = `-1` `1000,4000` `200` `-1,-1,10000,-1`

CSV detailing the number of instructions to skip from the corresponding trace. Skipped instructions are not displayed at all, and do not create empty space. May be specified for separate traces, to synchronize, for example, the first instruction in one trace to the 500th instruction to the other.

-InstructionMaxes = `100000` `1000,40000` `200` `-1,-1,100000,-1`

CSV detailing the maximum number of instructions to accept from the corresponding trace. The filament view will take a significant amount of time to display longer traces: in general it is a good idea to limit the instructions queried to 100,000 or fewer. Entering a -1 here will allow the filament view to query for all instructions, but this is very hazardous as the filament view may fail or hang if given more than about 500,000 instructions.

-InstructionOffsets = `0` `0,300` `1000,0,1500`

CSV detailing the number of instructions by which to offset each trace.

Offsets move forward from the origin. As an example, if an interesting instruction occurs at instructions 1500 and 1950 in two different traces, -InstructionOffsets=450,0 will offset the first trace so this interesting instruction occurs at the 1950th instruction index in both traces.

Disasm and comms windows are synchronized to these offsets, so setting -FocusInstruction=1950 will set the cursor and all above windows to correspond to this point of interest.

-IDSIds = `-1` `-1` `-1,-1,-1` `40,-1` `5030`

CSV (one to four values). The IDS IDs in question. A value of -1 indicates no IDS.

-PulseFilaments = `false` `true` `false,false,true` `true,true,false,true`

CSV of booleans detailing whether or not to pulse the corresponding filament. Using a pulse picks out one or more filaments visually, making them more distinct from their neighbors.

-ShowDisassembly = `false` `true`

If true, will display the Disassembly Window.

-ShowComms = `false` `true`

If true, will display the Communications Window.

-ShowAnnotations = `false` `true`

If true, will display the Annotations on the filament.

-ShowMemory = `false` `true`

If true, will display the Memory View along-side the trace.

-OverlayTraces = `true` `false`

If true, the traces will occupy the same space. Otherwise, the traces will be spaced out.

-LargeDisasm = `false` `true`

Whether or not to fill the screen with the disassembly window.

Works with one or two executions; any more will likely cause disassembly lines to be truncated.

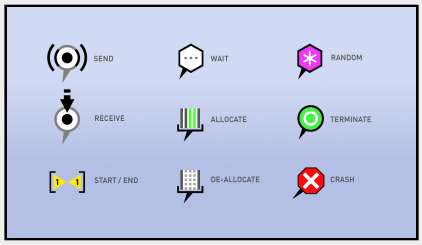
This will cause the filament to not display, as it would be mostly occluded anyway.

represented by the small, red and green Dot and Flag icons in the heading areas.

Dots are used for a Poll or Type 1 PoV, and flags are used for a Type 2 PoV.

- *The left execution is a Poll that was successfully handled by the CS. Both the Poll and the CS are green.*
- *The middle execution is a PoV that successfully proved a vulnerability against the CS. The PoV is green (it succeeded), and the CS is red (it failed to defend against the PoV).*
- *The right execution is a PoV that failed to capture a flag from the CS. The PoV is red (it failed) and the CS is green (it successfully defended).*

Polls and CS's should always have the same color. PoVs and CS's should have opposing colors.



These icons appear throughout the filament, and represent various things:

- *Crash*
- *Comms (Transmit/Receive)*
- *Vector35-Inserted Annotations (Start, End) (Deprecated?)*
- *Other Syscalls, etc. (Wait, Allocate, Deallocate, Random, Terminate)*

-MagicPageRange = 1128775680..1128779776

The range of addresses in memory where the Magic Page is.

-InstructionLength = 0.0001

The Z distance (depth) between points on the filament. May produce unexpected results.

-InteractiveCursor = false true

If this argument is found, the disasm and comms windows will not advance automatically. Instead, the user may click and drag the mouse near the primary filament to manipulate the cursor plane. This will also display the instruction index as a tooltip near the mouse. This argument is not intended for use in the VGS.

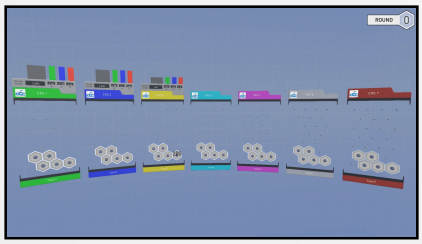
TOP

Arena View

Haxxis Package Name: ArenaView/ArenaView.json

Graphical representation of game events for the given round.

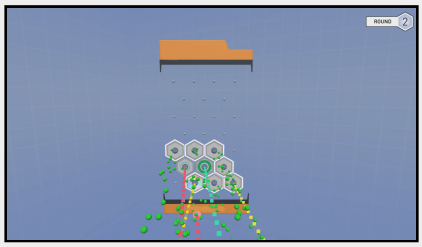
- By specifying TeamFocusId, this view acts as Arena Team View, focusing the camera on one team.
- Can optionally show just certain Teams and Challenge Sets.
- Can optionally show or hide PoVs and/or Polls.
- Some CS hexes will be gold. This indicates they are a special binary ("snack pack"). See screenshot, below.



Arena View, after all the PoVs and polls have been sent, in the middle of scores being tabulated.

Arguments:

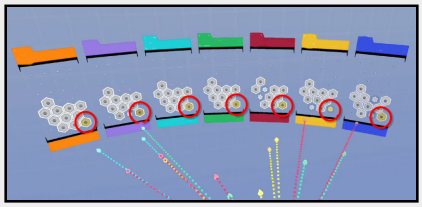
- *-Round=0*
- *-Anonymize*



Arena View showing one team fielding PoVs and Polls.

Arguments:

- *-Round=2*
- *-TeamFocusId=5*
- *-VisibleTeamIds=5*
- *-ShowPovs=true*
- *-ShowPolls=true*



Some CS's are gold (circled). This indicates that they are a special binary ("snack pack").

Required Arguments

-Round = 1 5 60

Integer. Identifies which round to generate this view with.

Optional Arguments

-ShowPolls = false true

Whether or not to show the Polls. Polls can make the visual busy.

-ShowPovs = true false

Whether or not to show the PoVs. Useful if, for example, polls are the focus of a given story.

-VisibleCsIds = all 2 4,7

The word "all", or CSV. Indicates which Challenge Sets to show in each team's hex grid.

-TeamFocusId = undefined 6

*Integer. If set, Arena View becomes **"Team View"**, focusing on the identified team's card.*

-VisibleTeamIds = 1,2,3,4,5,6,7 1 1..4

CSV or Range. Indicates which team cards to draw.

-ShowScoreTabulation = true false

Whether or not to show the score tabulation at the end of the sequence.

-TeamComparisonView

(No arguments) If present, then the teams specified with -VisibleTeamIds will be displayed, side by side, with static choreography.

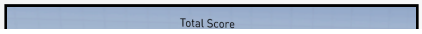
TOP

Line Graph

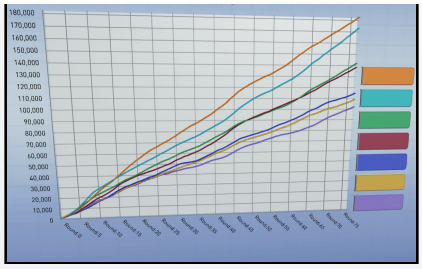
Haxxis Package Name: ScoreGraph/GeneralizedLineGraph.json

A heavily configurable line graph.

- Can show aggregate scores for each team over all rounds.
- Can show individual scores for one challenge set.
- Performs arbitrary operations per round and across multiple rounds.
- X axis (horizontal) is always round, but Y axis (vertical) and Z axis (series) are configurable.
- Frequently used to show total score over rounds.



Required Arguments



Required Arguments

-Rounds = `3..8` `5` `1,2,7,8`
The set of rounds to display. This argument constructs the X axis of the graph. Bear in mind that this does not support keywords, so one cannot specify 'All' here.

-YAxis = `Total_Score` `Availability.Total` `Availability.Performance.Total` `Availability.Performance.Execution_Time` `Availability.Performance.File_Size` `Availability.Performance.Memory_Use` `Security.Total` `Security.Consensus` `Security.Reference` `Evaluation.Total`
The data element to select from each scoring entry. These are case-sensitive.

Optional Arguments

-Teams = `All` `4` `1,2,5,7` `1..4`
The set of team IDs to draw information from. Note that this is not necessarily the same as the Z axis. If you'd like to display information about all CSs for a single team, this is where you'd specify that team.

-CsIds = `All` `49` `1,2,5,7` `5..9`
Inclusion list of CS IDs to use for data tabulation.

-AnnotatedRounds = `21` `15,26,45`
For the given list of round number(s), if any, will display a vertical bar in the graph at those round position(s); used to call attention to particular rounds in the graph.

-ShowAnnotatedRoundHeadings = `true` `false`
If true, the round annotation markers will have the label 'round N' at the top.

-ZAxis = `Team_Id` `CsId`
The data element to select from each scoring entry to create the Z axis, or series. These are case-sensitive. Each one must be a distinct value to produce useful data. Right now we can't use conditional statements for series, and we cannot alter the Y axis per series (such as in the desired case showing separate lines for A, S, and E scores for a given team).

-ZAxisLabelValue = `VALUE` `Crs_Name` `Challenge_Shortname`
The data element to select from each series for use as its series label. These allow us to display something like CRS names or challenge set shortnames instead of team and CS ids. The VALUE keyword uses whatever axis was selected for Z-axis above.

-PerRoundOperation = `Sum` `Average` `Min` `Max`
This describes how to combine the X and Z axes into a single Y axis value. Each distinct combination of round and Z-axis yields one or more data samples. For example, with z-axis=Stage_Order the data is reorganized by round and by team. Team 4, round 8 describes a collection of scores entries (in this case, one for each CS active in that round). If we've set yAxis=Total_Score the package selects the total score from each of those score entries. This PerRoundOperation then combines the Y axis samples into a single Y axis to display, pursuant to the CrossRoundOperation.

-ZAxisLabelFormat = `{0}` `Team {0}`
C# style format string used to construct the per-series labels. Each one is fed the label value (see below) it represents as a string. Other examples might include Team:{0}

-YAxisLabelFormat = `{0:N0}` `{0}` `{0} Units`
C# style format string used to construct the Y axis labels. Each one will be given an argument for the Y value of the grid line they'll represent.

-XAxisLabelFormat = `Round:{0:n0}` `{0}` `{0} Units`
C# style format string used to construct the round labels. The format string will always be given an integer-form round number.

-CrossRoundOperation = `Value` `Accumulate` `Diff`
This operation is performed across each final Y axis before sending it for display. The operation proceeds from the lowest round to the highest, so Accumulate and Diff will show the running totals or differences from round to round. 'Value' presents the obtained Y values without modification.

-MinDisplayRounds = `10`
If the input rounds encompass less than this number, extend the graph size to include at least this many rounds. For example, with Rounds=1..4 and MinDisplayRounds=10 the graph will show data on rounds one through four, but leave empty space for rounds five through eleven.

-UseZeroMinimum = `True` `False`
If this is true the vertical axis of the graph will always include zero. If not, the graph is sized to the minimum and maximum values present in the data.

-IdealNumberOfYMarks = `14`
Same as above, but for horizontal grid lines.

-IdealNumberOfXMarks = `18`
The graph chooses the spacing of vertical marks by minimizing the difference between this number and the number of marks it would generate for each interval. Giving it a smaller number will prioritize it towards less grid lines, giving it a larger number will prioritize towards more.

-DefaultLineColor = `0.7,0.5,0.1`
If a line corresponds to per-team data (using Stage_Order as the z axis) the lines will be colored according to their team. If not, use this string as an RGB color for all lines.

-GraphLabel = `"Total Cumulative Score"`
Draw this string as a world-space title across the top of the line graph.

-WipeGraph = `false` `true`
If true the lines within the graph are revealed over time rather than immediately, wiping on from left to right.

-WipeDuration = `10` `2` `20`
Defines how long the wipe takes (if used) in seconds.

TOP

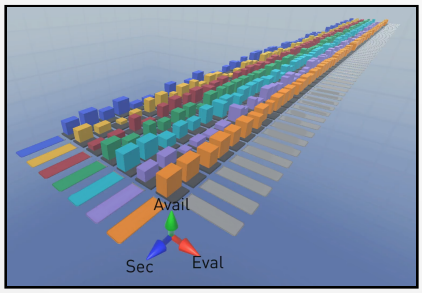
Cube View

Haxxis Package Name: GeneralizedIsoGrid/GeneralizedIsoGrid.json

A five-axis graph: X and Y in space, with each cube representing three values via its scale in X, Y, and Z.

- Draws positional axes from distinct integer data like Round, CsId, or TeamID

- Constructs cell sizes from float data like total score, security total, or memory use
- Frequently used to show A,S,E scores across various services
- Constructs a chart axis display with configurable axial labels



Required Arguments

-Round = 0..10 5 1,2,7,8
The set of rounds to display. Bear in mind that this does not support keywords, so one cannot specify 'All' here.

Optional Arguments

-XAxis = Stage_Order CsId Round_Number
The data element to use as the major X Axis. Is case sensitive. All entries with the same major axes will be aggregated together into a single cell. For example, if xAxis=Stage_Order and zAxis=Round_Number on Rounds=0..10, the cell (3,2) will display the average of the scoring data for team 3 in round 2. Note that while this field can accept Team_Id it is preferable to use Stage_Order instead, as it will produce the same results but will also respect stage sorting.

-XAxisLabelValue = VALUE Crs_Name Challenge_Shortname
The data element to select from each series for use as its series label. These allow us to display something like CRS names or challenge set shortnames instead of team and CS ids. The VALUE keyword uses whatever axis was selected for Z-axis above. As a common example, if xAxis=CsId it makes sense to use XAxisLabelValue=Challenge_Shortname and XAxisLabelFormat={0} to display the challenge shortnames instead of the CsId. As a second common example, if xAxis=Stage_Order it makes sense to use XAxisLabelValue=Crs_Name and XAxisLabelFormat={0}.

-Teams = all 4 1,2,5,7 1..4
The set of team IDs to draw information from.

-CsIds = all 49 1,2,5,7 5..9
Inclusion list of CS IDs to use for data tabulation.

-XAxisLabelFormat = {0} Round: {0} Cs: {0}
C# Label Format string used to construct the major X axis labels. Will be fed the result of the XAxisLabelValue below.

-ZAxisLabelValue = VALUE Crs_Name Challenge_Shortname
The data element to select from each series for use as its series label. See XAxisLabelValue above for more information.

-ZAxis = CsId Stage_Order, Round_Number
The data element to use as the major Z axis. Is case sensitive. See XAxis above for more information.

-ZAxisLabelFormat = {0} Round: {0} Cs: {0}
C# Label Format string used to construct the major Z axis labels. Will be fed the result of the ZAxisLabelValue below.

-ReverseXAxis = False True
Whether or not to reverse the order of elements along the major X axis.

-ReverseZAxis = False True
Whether or not to reverse the order of elements along the major Y axis.

-CellXAxis = Evaluation_Total Total_Score Availability_Total

-CellXAxis = Evaluation.Total Total_Score Availability.Total Availability.Performance.Total Availability.Performance.Execution_Time Availability.Performance.File_Size Availability.Performance.Memory_Use Security.Total Security.Consensus Security.Reference Zero One

The data element to select from each scoring entry to produce the x axis. These are case-sensitive, and represent information selected from each combination of major X and Z axes. Information within each cell is averaged to provide a single display value. For example, if xAxis=Round_Number, zAxis=CsId, and CellXAxis=Total_Score, the x dimension of each cell will represent the average of all teams' submissions on a given challenge set in a given round. Zero and One are static values.

-CellYAxis = Availability.Total Total_Score Availability.Performance.Total Availability.Performance.Execution_Time Availability.Performance.File_Size Availability.Performance.Memory_Use Security.Total Security.Consensus Security.Reference Evaluation.Total

The data element to select from each scoring entry to produce the y axis. These are case-sensitive. See CellXAxis above for more data.

-CellZAxis = Security.Total Total_Score Availability.Total Availability.Performance.Total Availability.Performance.Execution_Time Availability.Performance.File_Size Availability.Performance.Memory_Use Security.Consensus Security.Reference Evaluation.Total

The data element to select from each scoring entry to produce the z axis. These are case-sensitive. See CellXAxis above for more data.

-CellXMax = -1

The visual maximum range of a cell's x dimension. If this is set to 4 and a cell has an x value of 3, the visual cell produced will be 3/4 as wide as its base. If the default value of -1 is used, the package instead computes the maximum of all cells along this dimension and uses that as the visual range.

-CellYMax = -1

The visual maximum range of a cell's y dimension. See CellXMax above for more information.

-CellZMax = -1

The visual maximum range of a cell's z dimension. See CellXMax above for more information.

-GimbalXLetter = X(E) Evaluation Eval E

The text to appear on the X axis on the axis labels at the origin of the graph. Override whenever a different cell x axis is used.

-GimbalYLetter = Y(A) Availability Avail A

The text to appear on the Y axis on the axis labels at the origin of the graph. Override whenever a different cell y axis is used.

-GimbalZLetter = Z(S) Security Sec S

The text to appear on the Z axis on the axis labels at the origin of the graph. Override whenever a different cell z axis is used.

-FrameBoundsChoreo = False True

If true, modifies the choreography to sweep down the length of the visual space, front to back, rather than following its usual pattern.

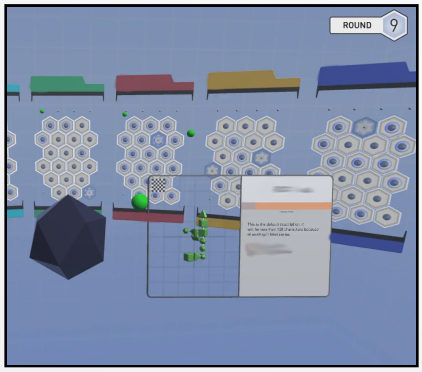
TOP

Arena View Intro

Haxxis Package Name: ArenaView/ArenaViewIntro.json

Visualizes the removal and introduction of challenge sets for the given round.

- If the generated video is very short and shows no CS's being added or removed, this implies that there are no CS's being added or removed for that round. Disregard such videos.
- See CS View for details on the CS Card.



Required Arguments

-Round = 2 60

Integer. The round for which to visualize challenge set changes.

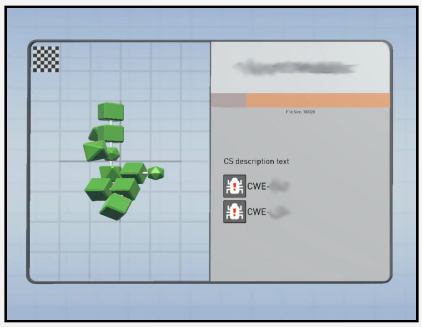
TOP

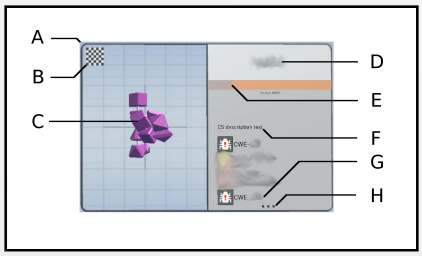
CS View

Haxxis Package Name: CsView/SingleCsView.json

Challenge Set View. Generates visuals based on attributes of a Challenge Set; including an abstract model, a list of CWEs, and a breakdown on binary composition.

- Note, this uses Challenge Set IDs, *not* Binary IDs.
- A gold-colored heading indicates that this is a special binary ("snack pack"). See screenshot, below.





Breakdown of the CS View:

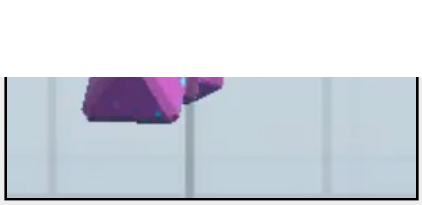
A. The entity referred to as the CS View
B. Entropy - The higher the entropy, the more dense the checkerboard (more checkers).
C. 3D abstract of the Challenge Set, based on a variety of properties. See next image for details.
D. Name of the Challenge Set. This background will be gold if it is a special binary ("snack pack").
E. Breakdown of the file sections making up each binary by size.
F. Description of the Challenge Set.
G. One known vulnerability.
H. Indicator that there are more known vulnerabilties than listed.

Required Arguments

-CsID = 10
Integer. The Challenge Set ID to visualize.

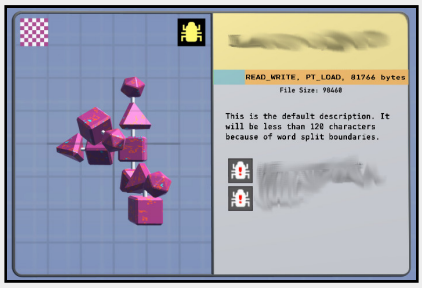
Optional Arguments

-TakeScreenshot
(No arguments) If present, the choreography will not record a video, but simply produce a single screenshot off the image.



3D Abstract of the Challenge Set.

- *The color is derived from a bitstream form of the shortname. Challenge sets with similar early characters in their shortname will have similar primary colors.*
- *The orientation of the primitives is determined by a bitstream form composed of a combination of the shortname and the file hash of the original binary. Challenge sets with similar shortnames but different file hashes will have very slightly different orientations.*
- *The flecked texture on the primitives is produced from entropy and opcode histograms. The system sorts all opcodes (including parameters) by the opcode string and uses them to generate a color spectrum. It then picks the most common few opcodes (commonly the most common two, but can be as few as one or as many as four) opcodes as its distinct color rings. Changes in the frequency of opcodes will thus produce large changes in the flecking colors, while similar opcode histograms will produce similar flecking colors. The shape and frequency of the flecking pattern is instead based on a binsets entropy. The color rings chosen above are distributed topographically based on a perlin noise function generated from the binsets entropy. Higher entropy will produce a larger number of sharper bands, while lower entropy will have a small number of large, smooth color bands.*
- *The overall size of the shape is determined by the file size of the binary. This expands logarithmically, currently with respect to log(1.7) (so an abstract with n+1 components has about 1.7 times as many bytes as an abstract with n components). Note that the visual occlusion of the shape, or the visual size occupied, is not related to the size of the file: each abstract is resized to fit within the available visual space, so the size of each component is not significant.*



This is an example of a gold-colored CS heading, indicating that this is a special binary ("snack pack"). There is also a gold "bug" icon.

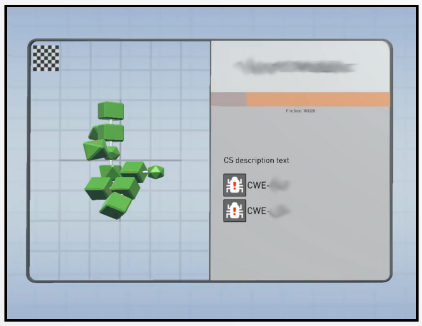
TOP

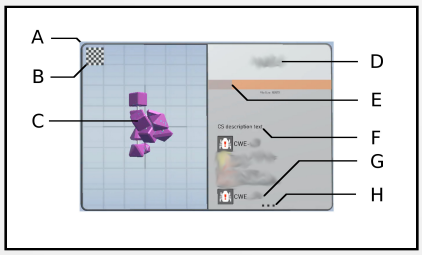
Binset View

Haxxis Package Name: RcsView/SingleBinsetView.json

Binset View. Generates visuals based on attributes of a Binary Set; including an abstract model, a list of CWEs, and a breakdown on binary composition.

- Similar to the CS view in function.
- Note, this uses Binary Set IDs, *not* Challenge Set IDs.





Breakdown of the CS View:

A. The entity referred to as the CS View
B. Entropy - The higher the entropy, the more dense the checkerboard (more checkers).
C. 3D abstract of the Challenge Set, based on a variety of properties. See next image for details.
D. Name of the Challenge Set.
E. Breakdown of the file sections making up each binary by size.
F. Description of the Challenge Set.
G. One known vulnerability.
H. Indicator that there are more known vulnerabilties than listed.

Required Arguments

-BinsetID = 160
Integer. The Binary Set ID to visualize.

Optional Arguments

-NameOverride = SHORTNAME Galactica's patch of this service Reference patch for CS 144 Inefficient patch by JIMA
String argument that allows an override of the normal title display. If this arg is the default SHORTNAME the card will display the challenge shortname instead.

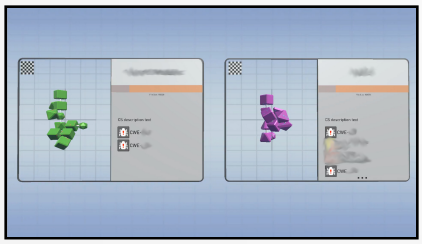
TOP

Binset Comparison View

Haxxis Package Name: RcsView/CompareTwoBinsets.json

Challenge Set Comparison View. Same as Binset View, but compares two rcs' side by side.

- Note, this uses Binary Set IDs, *not* Challenge Set IDs.



Required Arguments

-CsIds = 1,10

CSV. Two values only. The Challenge Set IDs to visualize.

Optional Arguments

-DisplayStrings = SHORTNAME,SHORTNAME Galactica's patch,Rubeus's patch Reference binary for CS 144,Reference patch for CS 144 SHORTNAME,Inefficient patch by JIMA

A CSV of two Strings that allows an override of the normal title display. If the value is the default SHORTNAME the card will display the challenge shortname instead.