Real-Time Collaborative Network Monitoring and Control Using 3D Game Engines for Representation and Interaction

Warren Harrop
Centre for Advanced Internet Architectures
Swinburne University of Technology
Melbourne, Australia
wazz@swin.edu.au

Grenville Armitage
Centre for Advanced Internet Architectures
Swinburne University of Technology
Melbourne, Australia
garmitage@swin.edu.au

ABSTRACT

Identifying and reacting to malicious or anomalous IP traffic is a significant challenge for network operators. Automated real-time responses have been simplistic and require followup actions by technically specialised employees. We describe a system where off-the-shelf 3D game-engine technology enables collaborative network control through familiar 'interaction' metaphors by translating network events into visually-orthogonal 'activities'. Anomalous behaviour is targeted by the managers-as-players using in-game techniques, such as 'shooting' or 'healing', resulting in defensive actions (such as updates to a firewall's access control list) being instantiated behind the scenes.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*

General Terms

Management, Measurement, Security, Human Factors, Performance

Keywords

Enterprise networks, Network visualisation, Network control

1. INTRODUCTION

Network operators are continually challenged by the task of identifying and reacting to, anomalous (and potentially malicious) Internet Protocol (IP) traffic entering, traversing or leaving their systems. Enterprise networks are particularly concerned with traffic probing for weaknesses in local hosts and servers - especially when the probes originate from other hosts within the enterprise network itself. All network operators share a concern with denial of service (DoS) attacks - traffic patterns deliberately constructed to congest vulnerable links, routers or end-systems.

Reaction to anomalous network events (such as updating firewall rules, re-routing traffic, etc) is often labour-intensive and does not occur in anything close to real-time (except in simple, clearly defined computer controlled scenarios). As networks carry more and more mission critical data, the need for collaboration on these network management decisions becomes increasingly important. The implementation of the wrong network management decision can possibly have an impact as devastating (if not more devastating) than the original problem. Typically the detection, interpretation and reaction process requires time from multiple staff with relatively high skill levels - time and skills that might be more cost-effectively utilised.

In this paper we describe an unconventional technical approach for traffic monitoring, identification and control that arises from two basic premises: we wish to allow network administrators the option of real time computer assisted collaboration on anomaly monitoring in enterprise networks and we believe the human mind is better than computer algorithms at pattern recognition.

Our inspiration comes from previous work on network activity visualisation [40, 20, 34], virtual-world metaphors for interacting with computer process space [25, 2, 16] and virtual world collaboration systems [38, 31]. We create a virtual world where network events are rendered in real-time as visually-orthogonal 'activities' and network elements are controlled using metaphors for interaction that seem 'natural' and familiar to the human operators.

In recent times, people have explored the use of darknets[1] [41] and greynets [35] to passively monitor for the tell-tale signs of network scans in progress. In this paper we make use of the greynet to provide passively collected network intrusion information.

For example, monitored IP addresses may be represented by avatars that jump and spin in response to network events such as port scans. Interacting with a spinning or jumping avatar would be translated into an appropriate network reconfiguration (for example, temporarily updating an internal firewall's access control list). With appropriate translation of network events into the virtual world and translation of virtual-world 'interactions' back into network reconfiguration events, we can reduce the network-specific training required for staff charged with identifying and suppressing anomalous traffic across an enterprise network.

[1]Some in the networked-applications community have used 'Darknet' to mean 'A subversive, underground and illegal content distribution network' [23]. However, we use it here in the usual IP networking sense.

Key to practical implementation is our use of off-the-shelf multiplayer 3D game-engine technology. With modification, modern game-engines enable the creation of computer-controlled avatars ('bots') whose behaviours are tied to monitored network events, and whose reactions to being 'shot' or 'healed' (common in-game metaphors for interaction) can be translated back into network reconfiguration events behind the scenes. We have developed our first prototype using the open-source Cube [3] game engine.

An advantage of multiplayer game-engines is they are optimised for distributed involvement. There may be multiple network-managers-as-players logged into the virtual world at the same time, from different physical locations, viewing different parts of the monitored network. In addition, rules for interaction are easily configured, so that multiple participants can be required to 'shoot' or 'heal' a particular avatar before a network reconfiguration is instantiated (analogous to military nuclear missile launches that require multiple keys to be turned simultaneously).

A major contribution of this paper, is our evaluation of how we might most effectively map various IP network events into visually orthogonal avatar behaviours - crucial for enabling the human minds viewing these behaviours to be our optimal pattern recognition tool. For example, a network host address might be a pyramid shape that rotates with angular velocity proportional to the rate of inbound or outbound IP packets and whose size is proportional to the spread of port numbers or IP addresses in those packets.

This paper continues in section 2 with a review of related and existing work. Section 3 discusses the rationale for our particular work, followed by a detailed discussion of representation and interaction metaphors in section 4. Collaboration support is discussed in 5 and our choice of 3D game engine is outlined in section 6, followed by a description of our prototype - its design and operation - in section 7. Section 8 touches on future work before we conclude in section 9.

2. EXISTING AND RELATED WORK

Before describing our particular model and prototype experience, we first review some representative examples of related work on network monitoring, visualisation and control. We then review previous research into collaborative systems relevant within this area.

2.1 Network monitoring

A range of network monitoring techniques exist today. These include flow-based tools that monitor variably aggregated traffic patterns between third-parties, pattern-seeking tools watching for anomalous activities and enterprise-based tools that detect unsolicited attempts at communication to unused network locations.

Flow-level views of network activity (what hosts are communicating with other hosts and at what speeds) are provided by a variety of flow-export solutions, such as Netflow [11] and IPFIX [27]. These statistics can be collected and aggregated by tools such as SNMP [24] and Hewlett Packard's "HP OpenView" [6] to provide graphical representations of network statistics. However, these tools provide only coarsely-grained views into the dynamic state of a network.

Network Intrusion Detection Systems (NIDS) provide a somewhat more complex, packet-level monitoring service, as they can listen passively and promiscuously to network data flows for events that constitute an anomaly. NIDS can be roughly categorized into three types: event based, darknets and honeypots [5]. Event-based detection relies on either pattern matching, or statistical analysis of low level network data (for example, as implemented by Snort [15] and Bro [1]). Alerts are logged and optionally sent directly to network operators, when conditions are met that match connection patterns indicative of known network-level attacks.

Darknets [17] (also known as network telescopes [42], Internet motion sensors [21] and black holes [29]) are large contiguous portions of unused (but still reachable) IP address space. This IP address space is passively monitored and although it contains no legitimate hosts, will still receive inbound packets. These packets can represent active scans across the darknet space itself (for example, from naive worms seeking exploitable hosts) and backscatter [41] from attacks occurring elsewhere on the Internet. By looking at the return address on the packets received by a darknet, the network administrator can determine what network hosts are scanning the network and (in the majority of cases) are likely looking for other hosts to infect with malware. With this information, the network administration team can then make decisions on firewall changes if they are required.

Greynets [35, 36] modify the darknet concept by monitoring unused IP addresses diffused amongst 'normal' active IP addresses in an operational enterprise network. Detecting worm scans within an enterprise network is particularly useful for detecting and tracking worms that have taken up residence on 'trusted' hosts inside the enterprise's boundaries. For our prototype we have chosen to use a greynet's output to provide input into our visualisations.

2.2 Visualisation

All the monitoring methods mentioned above provide raw data as to a network's state. However, to be useful we must appropriately represent and interpret the data for human or computer consumption. Visualisation of network activity has been demonstrated using a variety of 2D and 3D forms.

Some projects have focused on visualising physical network topology or Internet route state (from Cheswick et al. [26] to CAIDA work [19]). However, our focus is on visual representation of traffic flowing through (or ending in) an operational network - ideally in a way that enables real-time recognition of problems or threats.

The simplest and most commonly used network visualisations are 2D representations of a single network metric in graph form. For example, the Multi Router Traffic Grapher (MRTG) [9] (and its "sibling" software Round Robin Database, RRDtool [14]). More advanced tools such as Nagios [10] monitor network hosts and services for unexpected outages and can alert network operators through various alarms and present metrics via a number of different graphical representations. Further research into the 2D visualisation of network metrics and IDS output has resulted in proof-of-concept software being developed in parallel, as with Conti and Abdulla [28], Ball et al. [22], Lakkaraju et al. [39] and Goodall et al. [33].

Novel ideas have also emerged in the realm of 3D visualisation. Abel et al. described Cybernet in 2000 [20] where they proposed the creation of 3D virtual worlds into which 3D objects would represent the state of monitored metrics (e.g. networks or filesystems). A successor to this work by different researchers in 2004 proposed the concurrent representation of independent network metrics using visually orthogonal 3D objects in fully immersive 3D space [34]. A unique aspect of this work [34] was the proposed use of dynamic behaviour, such as rate of spin or oscillation, to augment the visual distinctiveness of 3D objects that represented network entities or metrics.

In early 2004 the "Spinning Cube of Potential Doom" (SCPD) [40] demonstrated a practical approach by taking darknet data and placing dots within a three-dimensional cube, whose axis related to source IP address, destination IP address and destination port number. Dots representing these potentially malicious packets had a constrained lifetime, making network scans visible to the eye (vertical lines represent a port scan on a single host, flat two-dimensional planes appear for port scans upon multiple continuous host IP addresses, other port scans trying to avoid detection produce spiral "Barber Pole" like patterns). Work in 2005 by Komlodi, et al. [37] and Oline and Reiners [43] further explored the representation of complex traffic relationships between network entities using spatial orientation and linkage of simple images (dots, lines and balls) in 3D space. Other researchers have added 3D audio stimuli (to provide spatially-significant information [44]) and force-feedback stimuli (haptic integration [47]) to their representations of network state information.

2.3 Interaction

A number of projects deserve recognition because they stepped beyond visualisation and explored mechanisms for interaction with the systems being represented in 3D space.

A direct inspiration for our own work is psDoom [25] - a virtual-reality front-end to the unix process inspection program 'ps'. psDoom is a modification to Doom, the classic 3D first person shooter game. You 'play' psDoom on a unix-like machine and are presented with rooms and corridors populated by monsters. In regular Doom you are expected to interact with the monsters by shooting them (it is a simple game). However, in psDoom every monster represents a different process in the underlying unix system's process table. Using the game's metaphor for interaction, a player can 'shoot' a monster to reduce the priority of the associated process. When the monster is killed, the associated process is likewise killed. (The first person shooter metaphor was also used by the Brutal File Manager [2] - a simplistic Java3D application that allows users to see, and 'shoot', files in their underlying file system.)

More recently Sun Microsystems' open-source, Java-based "Project Looking Glass" seeks to "...break two boundaries – the 2D-ness of the current desktop environment and the way the desktop environment evolves." [16]. A user's desktop is rendered in 3D form, and the user interacts with the desktop objects in a manner consistent with their apparent position in the virtual 3D space.

2.4 Collaboration

Although all of the previously mentioned research efforts have made significant contributions and advances in different areas, none have focused on collaboration as a central issue for development. The great importance of collaboration in making administration decisions is evident. Goodall [32] shows the importance of collaboration in the field of network intrusion detection and finds "...how apparent solitary security work is richly collaborative both in the learning strategies and in the mundane and exceptional performance of the work" through field study of security experts.

Our goal is to create a virtual environment building on existing 3D game engine technology, in which network administration collaboration can take place. A number of works in the CSCW field are relevant to our goals. As long ago as 1992 Takemura and Kishino [48] described the idea of highly visual, collaborative, virtual workspaces and interactive systems that can be manipulated by two users simultaneously.

The idea of leveraging collaborative games to provide not only recreation and social interaction, but perform more traditional work, has also existed since MOOs (MUD Object Oriented) and MUDs (Multiple User Dimension, Multiple User Dungeon, or Multiple User Dialogue) [49]. The potential for games to be serious tool of teaching and working is still being actively researched [45, 46].

Closer to our goals, direct leveraging of a 3D game engine has recently been accomplished by Kot et al. [38] to perform code visualisation tasks. Through modification (of the now open source) Quake III engine [7] multiple 'players' can enter a virtual world where source code is represented as virtual objects and users can jointly view contents of files and 'point' to areas of interest. In 2001 the Half Life 1 engine was used to visualise workshop outcomes and allow discussion of these collaboratively in real time, both by attendees locally and connecting via the Internet [31].

3. RATIONALE FOR OUR WORK

Our approach for traffic monitoring, identification and control arises from two basic premises: we wish to allow the operational staff of IP networks to collaborate in real time on solving anomalous network behavior and we believe the human mind is better than computer algorithms at pattern recognition. We leverage modern 3D game engine technology to cheaply and conveniently extend the ideas articulated in [34] and inspired by [25].

When a NIDS provides automated monitoring and control, there are three intrinsic steps - detection (of network packets), interpretation and reaction. The middle step - interpretation - is particularly problematic for computers. When humans are introduced, we require four intrinsic steps - detection, representation, interpretation and reaction. The challenge is to perform representation in a manner that enables real-time assessment of network conditions by a human and real-time interaction with the network (the reaction step). By choosing the right metaphors for representation, our contention is that a 3D "world" can be created in which useful network administration can take place by relatively untrained individuals using intuitive interaction metaphors.

The immersive environment we define, can easily be implemented on common "off the shelf" PC hardware. Multiplayer 3D games have pushed consumer PC and video card technology well beyond what we required for modest network visualisation. The sheer scale of the game market ensures powerful 3D video cards and software interfaces such as OpenGL [13] and DirectX [8] are a trivial financial cost. Even entry-level 'home' PCs come with sufficient 3D video technology for our purposes. Finally, many modern 3D game engines provide cheap, flexible and reconfigurable networking and software platforms for instantiating our proposed method for representation of and interaction with, the net-

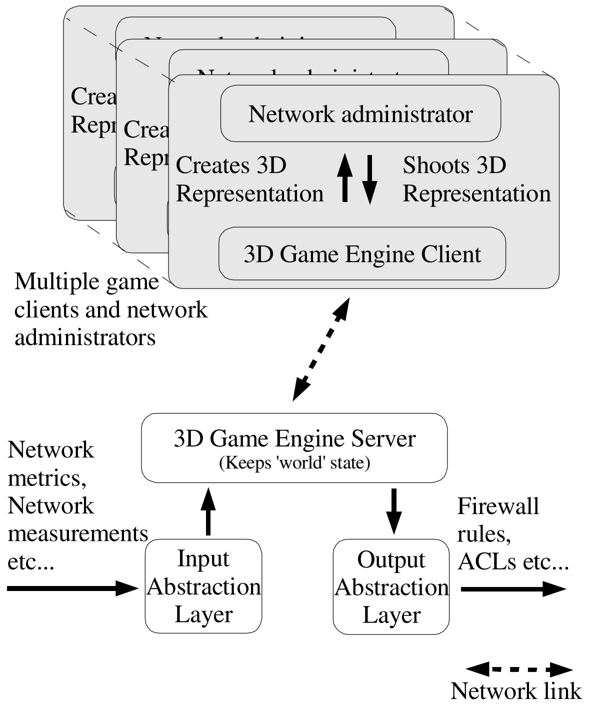

Figure 1: Collaborative network monitoring and control with a 3D game engine client/server setup

work by multiple connecting clients. Hence we propose the system shown in Figure 1 - network state is deterministically mapped to 3D entities controlled by a modified game engine server, the network administrator (as 'player') views the representation of network state via the game client, interacts with objects inside the game world using in-game metaphors and the interactions are deterministically mapped to actual network reconfiguration activities.

4. METAPHORS FOR REPRESENTATION AND INTERACTION

Before expanding on our system prototyping experience, we discuss further our central ideas regarding appropriate representation of dynamic network state.

4.1 Visual metaphors and visually orthogonal

To simplify our human operator's instinctive pattern recognition process we represent network state using artificial versions of familiar objects and shapes from the real world. In order to concurrently represent multiple network characteristics inside our 3D virtual world, we must map independent network state variables to visual representations that are clearly and unambiguously distinct when viewed simultaneously on screen - what [34] termed 'visually orthogonal'. Visual orthogonality increases the human operator's ability to differentiate concurrent network events in the immersive environment.

A single object may have shape, size, colour, texture, and dynamic behaviours such as spin and/or oscillation about a fixed point. An example of 'visually orthogonal' attributes for a pyramid shape might be spin rate and up-and-down oscillation - it is unlikely that variations in spin rate around an axis would be mistaken for 'bouncing up and down'. Conversely, colour and spin rate would not be visually orthogonal for a smooth, uniformly coloured sphere - it would be exceedingly hard to judge changes in the sphere's spin rate without some visual cues that it was spinning.

Some of these characteristics lend themselves naturally to representing quantised values whereas others lend themselves to representing a continuum. For example, mapping IP packet type (TCP, UDP, ICMP) to object size would not allow the three packet types to be discerned easily, as absolute object size within a 3D space requires much more interrogation to determine distinct values than, for example, colour. In addition to visual orthogonality, our choice of metaphors should also have an intuitive (or easily learned) relationship to the network metric the metaphor represents.

Table 1: Visual metaphors to network metrics map

Visual Metaphor	Network Metric
Location	IP address, port number
Shape	Representation of object type (subnet, host or connection)
Size	Time aggregate of unique connections
Colour/Texture	Content type
Rotational Velocity	Throughput
Oscillation about a fixed point	User defined alert

4.2 Mapping of Visual Metaphors to Network Metrics

Table 1 summarises what we believe to be a reasonable first-pass at mapping visual metaphors to networking metrics.

We propose to use three types of visually orthogonal primitive shapes to represent three levels of network aggregation. Entire subnets (sub-networks eg 192.168.0.n) are represented by cubes, hosts are represented by pyramids and horizontally oriented cones represent individual connections or flows. In a game-engine implementation these objects are instantiated by in-game avatars. Each avatar's location within the virtual world corresponds to their IP address range, IP address or port number; for cubes, pyramids and cones respectively.

The colour of a particular avatar represents the "type" of item it is, in the case of cubes (subnets) color denotes what "type" of subnet it is (for example, a collection of enterprise servers, a darknet, end user hosts, etc). Similarly, for end network hosts, colours represent the "type" of host ie. workstations, darknets, servers, VoIP phones etc. Connection avatars (cones) have three colours to represent UDP, TCP and ICMP. (Specific choice of colours is an implementation issue that must keep in mind the presence of colour blindness in the population [50], the inability of most people to differentiate subtle variations in colour and the fact certain colours and textures can appear ambiguously similar.)

To represent the number of connections and bandwidth consumption by a particular object, object size and spin rate are used as visually orthogonal metaphors. As the number of unique connections to an object increases, so does its size. As the bandwidth consumed by the object increases, it begins to spin faster. Thus a large pyramid object spinning slowly represents a host with a large number of unique connections, but a low overall bandwidth consumption. A small pyramid object spinning quickly represents a host with minimal connections but a large bandwidth consumption and a large object spinning quickly represents a host that most likely requires administrator attention.

Oscillation around a fixed point (a "jumping" type of motion) is one of the most attention gaining movements an avatar can make. For this reason oscillation has been chosen to represent user defined alerts. Even when somewhat subtle network metrics may have occurred, by placing user defined alerts upon them, they become extremely obvious. (In our prototype, described later, we use oscillation - jumping up and down - to augment the spin of an avatar.)

To aid simplicity of view, we propose representing network hierarchy through aggregation. At the highest level, avatars represent entire subnets (cubes), to further investigate the operation of a subnet, the user moves the virtual world camera into the subnet cube object. Inside the subnet cube all avatars are host objects (pyramids). Further moving into a host object reveals the objects representing the particular host's connections (cones). With this three tier system an overall view can be gained of the network, but it also allows for further detail to be gleaned when required. Where possible, similar metaphor mappings are used at each level of the hierarchy. (For example, the size of an avatar representing a subnet might be proportional to the number of connections entering or leaving the subnet. This would be consistent with the use of size to represent the number of connections in and out of an avatar representing a host.)

4.3 Nonlinearity of network metrics

Since our goal is to aid in human recognition of patterns across the network, we are not required to provide fine-grained and wide-range representation of network state. In addition, we almost certainly do not want to provide linear representation of network state. As a simple example, TCP and UDP ports technically run from 0 to 65535, yet in real-life traffic there will be significant clustering of port numbers in certain ranges (e.g the reserved ports range from 1 to 1023, and groupings around unreserved - but defacto well-known - port numbers). A linear mapping from port number to visual metaphor, would not be efficient.

To work around this issue, scales of space will generally have to be nonlinear and dynamically change their size to exaggerate the areas that require attention. Spin rate too should have a nonlinear scale, as past a certain velocity (that can be system defined), a fine grained inspection of bandwidth is no longer necessary or possible. The only knowledge required is that a network object is exceeding a predetermined limit.

To ease detection of network events that occur in milliseconds, all metrics are calculated based on sliding window averages across time. This prevents possibly sub-second events from being overlooked and gives a more realistic idea of network resource usage by "smoothing" out metrics that may burst on a moment to moment basis. (A future question is how to weight the averaged value to more strongly represent newer information over older information.)

4.4 Interaction Metaphors

Having established the visual representation of dynamic network state, we must enable the operators to act upon observed information and implement network changes.

We suggest three key interaction metaphors that would be familiar to most game-players: shoot (with gun), heal (with syringe) and fine-tune (with pliers). Each of these actions would be context sensitive. For example, shooting an end user host would naturally instantiate a firewall rule stopping that host from accessing the wider network. Shooting a greynet 'host' (a 'dark' IP address) would, more reasonably, instantiate a firewall rule aimed at blocking the host that was sending data to (or 'attacking') the greynet address.

While shooting represents firewall rule enforcement, the heal metaphor performs the opposite task, it is for backing out changes that have been made in error. The pliers are used as an alternative to shooting an avatar, using them might indicate "implement rate limiting" on a particular network object (incrementally increasing the effect as the pliers are 'used'). Again, the syringe can be used to "undo" its actions.

4.5 System Features

In addition to visual metaphors and interaction metaphors we also define a number of other broader system features needed to aid in the intuitive diagnosis of network issues.

The historical context in which a network event occurs can often be important in defining the event. Thus, the ability to move not only in three dimensions, but also in time, is of benefit. The ability to spool network events into a log and playback these logs (possibly at variable speed) from within the system, allows extra context for an event to be determined. It would be advantageous to be able to collaboratively achieve this, in a manner similar to [38] by locking the view of one player to another.

As an optional feature for more advanced users of the system, when the virtual camera of the immersive environment is moved close to an object, on screen display can be toggled to show detailed numerical information for the particular avatar under inspection. This allows for network metrics to be seen at a glance by even the unskilled eye, but also allows detailed metrics to be interrogated if required by an advanced user.

Although an important requirement, at this point we have not defined any specific custom security features for our system. Current game engines do implement various anti-cheat mechanisms, but none encrypt and/or authenticate client/server communications. Future versions of our systems may include their own methods, but for now we leave this important function to other protocols (eg IPSec) to provide a secure and authenticated communications method for our network metrics and control data to travel across.

5. COLLABORATION

When using our definitions, multiple administrators can enter a virtual environment at the same time. Multi-player functionality in modern networked games makes this trivial for us to implement. When we implement world changes (eg avatar movement due to changing network conditions) we do this at the game server. This then transfers these state changes automatically to all connected client machines using existing network code.

With multiple network administrators 'inhabiting' the 3D environment, the need may arise where administrators can be delegated different parts of the network - only authorised to implement changes within their particular sections. Administration rights can be created to satisfy any requirements of administrator delegation from "view only" to "full veto rights on any action".

By using the standard 'users and groups' administration model and allocating network objects within the world both 'read' and 'write' (or 'view' and 'interact') privileges allows different levels of access rights to be enforced. Most levels of access rights would be determined by physical location within the virtual world as this represents logical layout of the underlying IP network. Thus administrators may have limited 'areas' of the network for which they are responsible.

Having multiple administrators within the world introduces some advanced network control possibilities. Not only are two (or more) people now watching for anomalies within the network, the possibility of a incorrect or accidental network adjustment can be reduced. In certain circumstances it can be enforced (by setting levels of administration rights) that multiple network administrators are required to shoot an avatar before a network change is implemented. This way a certain number of votes (shots) are required to accumulate (over a relatively short period of time) for a network change to be implemented as a real 'live' network update.

6. LEVERAGING 3D GAME ENGINES

The input and output abstraction layers in Figure 1 implement network metric to visual metaphor mapping and interaction metaphor to network control messages. There are a number of different reasons for utilising existing "off the shelf" 3D game engines when implementing an immersive environment:

- Multiplayer 3D game engines already incorporate proven, optimised network code to support real-time, distributed involvement of multiple operators (players). The game engine code also meets our requirements for 'real-time' allowing us to see and send network metric and control messages within 10s of milliseconds (100s if the network administrator is located on another continent to the network they are administrating).

- 3D games provide us with high level tools with which to express our desired 3D representations and the virtual environment they will inhabit.

- 3D game engines handle all the complex 3D rendering in real-time on behalf of our network monitoring and control system.

- Many 3D game engines provide hooks to allow 3rd party modifications. These hooks allow us to take input from an external source and send output to an external system, simplifying the implementation of our network monitoring and control system.

We chose the Cube game engine [3] because the full source was easy to obtain, and it included an innovative in-game map-editor. Early versions of the well-known Doom and Quake series of games [7] have also had their full source code released (under the GNU Public Licence). Developer toolkits are available for other, more recently released games, such as Half-life 2 [18]. These toolkits allow a wide variety of game-play modifications without releasing the entire game-engine's source code.

7. PROTOTYPING

A prototype system was created implementing a large number of the features outlined above. Moving away from the generality of Figure 1, our prototype uses the input network metric of packet arrival into a greynet, the 3D game engine Cube was modified to implement the 3D world and the output from the system was an ACL (Access Control List) placed onto a cisco router.

Our experimental network's layout is shown on the right side of Figure 2. A Cisco router sat between our 'outside world' network and an internal enterprise network configured as 5 small subnets. Our greynet involved registering 5 greynet hosts in each subnet. The Cisco router's ACL was remotely configurable (using telnet from the game server host). Both the greynet packet sniffer and visualisation hosts were implemented on a single machine running FreeBSD 5.4 [4]. As a 'first person shooter' (FPS) the Cube game engine provides an interface where each player navigates the 3D world from the perspective of their character's eyes. We modified the Cube game engine to create our desired look and feel, and introduced new code to support the abstraction layers from Figure 1. In FPS games the player is generally represented (in the lower half of the screen) as carrying a tool or weapon for interacting with entities within the virtual world. The left side of Figure 2 shows our current player carrying a gun and surveying the 25 greynet 'hosts' (distributed over 5 subnets). The view also contains traditional in-game elements - such as "health", "armor", and "ammo" indicators - which are unused in this prototype.

7.1 Cosmetic changes to game engine

A custom map was created, simply a large "room" with a high ceiling. A raised "catwalk" runs around the outside wall of the map with stairs so the network administrator can move the virtual-world character to gain a bird's eye view of the visualisation. Movement around the world by the operator is achieved by the defacto FPS standard of 'a' 'w' 's' and 'd' keys to move laterally and mouse to control the direction of sight.

Cube's original "monster" characters were replaced with custom avatars (pyramids with a texture of red stripes, to allow rotation of avatars to be easily seen) for greynet host representations. The 'knight' avatar was retained, however, from the cube game as the most human looking avatar to represent other network administrators. As opposed to the multiple levels of hierarchy mentioned earlier, for sake of prototype simplicity, the "ground" in the world consists of various strips of texture; these represent the 5 different subnets to which the various greynet hosts belong.

7.2 Source changes to game engine

A number of source code modifications were also made to the underlying Cube engine. The Artificial Intelligence (AI) code was modified to make avatars receive their movement based on the greynet state. Other code was modified to detect when avatars have been shot and call an external helper script that acts as an abstraction layer to the Cisco router.

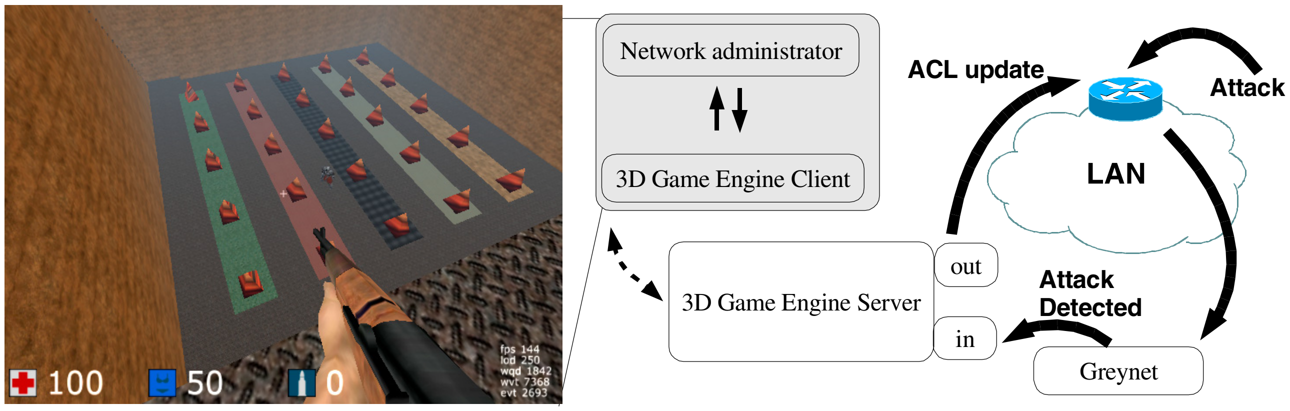

Figure 2: Virtual representation of the greynet's 25 addresses (left) and experimental network layout (right)

7.3 A day in the life of fully immersed network operators

The following sequence of events illustrates our prototype representing an active network scan (as summarised by Figure 5):

- Initially the greynet is idle (ie experiencing no ingress packets). Avatars representing the greynet hosts stay still.
- We launch an "attack" on the test network to represent malicious scan activity. An external host uses nmap [12] to send (TCP SYN) packets to port 445, linearly walking across one of the enterprise network's address space.
- An avatar begins jumping and spinning, indicating that a TCP SYN packet has been detected heading towards the associated greynet address.
- As nmap continues to scan across the subnet, the remaining avatars associated with that subnet also begin spinning and jumping (Figure 3).

An entire row of avatars jumping and spinning at once is easily detectable as out-of-the-ordinary network behavior. The network administrators can instantly see the extent of the network scanning - whether it is localised to one host, one subnet or large sections of the enterprise network.

The following steps illustrate how our network operators intervene:

- Due to each avatar's uncharacteristic dynamic behaviour the network operator infers that there is reason to intervene.
- Operators move into a position from which they can interact with one or more avatars using the available tool (in this case, the gun).
- Operators agree that network condition is genuinely anomalous and intervention is needed Operator 1 'shoots' an affected avatar.
- Within a short period of time, operator 2 'shoots' the same avatar (Figure 4).

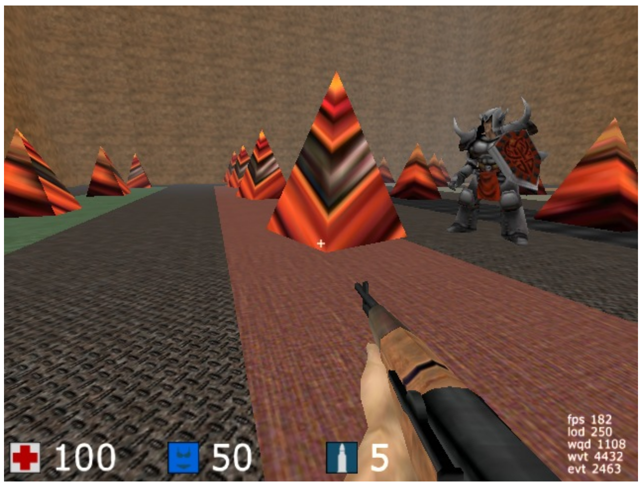

Figure 3: Spinning and "jumping" greynet host avatars signaling a network scan in progress - another administrator looks on

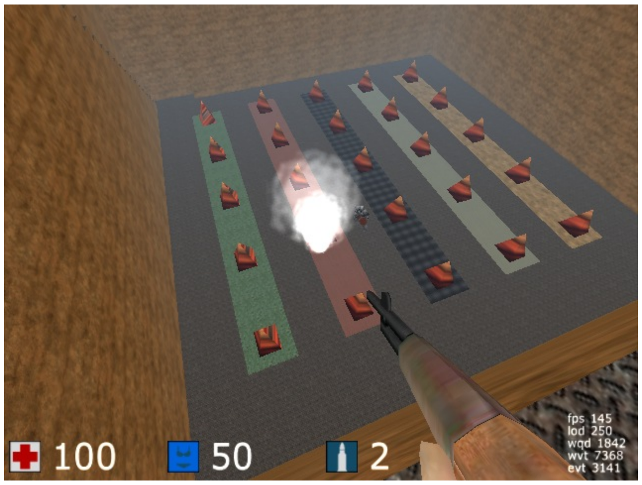

Figure 4: The second administrator taking a birds-eye view shoots a greynet host to place an Access Control List (ACL) entry

- The game-engine translates this into an ACL update on a Cisco router (from Figure 2), implementing a block on the address from which the inbound packets are arriving.

- All avatars stop their spinning and jumping as the network scan packets no longer enter our 'enterprise network'.

Future versions of our prototype will implement the additional interaction metaphors mentioned previously, such as using 'pliers' to instantiate rate-limiting in the boundary router rather than a complete block rule.

8. FUTURE RESEARCH

Our prototype implementation has a number of shortcomings and a short-term goal is to bring the prototype fully into line with our definitions. Once this has been completed, more comprehensive testing of the system can begin. This will be on a technical level (load testing, exploring scalability etc.) but also, at this time, we have only asserted what we believe to be intuitive metaphor mappings. To be fully rigorous, we must perform usability trials to discern the level of orthogonality of our chosen metaphors and their appropriate use within the system. A series of human factors tests with varying numbers of network administrators and varying test network conditions will constitute future work. During these tests we will fully explore the collaboration systems and possibilities of the software beyond the basic functionality we have at present.

It has become apparent during prototype development, when using greynets for enterprise attack detection, a standardised method for obtaining greynet state is needed. For greynets to become a common part of an NIDS on an enterprise network, the link between the specific method of implementation and method of monitoring must be severed. The Intrusion Detection Exchange Protocol (IDXP) [30] is defined in an expired Internet draft from the IETF. It presents as one possible method via which greynet interrogation could occur. In addition, we propose the creation of a MIB extension to allow greynets and darknets to be properly interrogated as to their state by SNMP [24].

9. CONCLUSION

In this paper we have been inspired by previous work (such as psDoom and the spinning cube of potential doom) that allow novel 3D interaction with a computer, normally achieved via 2D interaction or a command line interface. Previous work in network state visualisation has ranged from the very simple (graphs of single network metrics) to complex 3D world representations of multiple network metrics. None of this prior work, however, has approached with the implicit goals of allowing these worlds to be collaborated within, by multiple administrators and reducing the level of traditional IP networking knowledge required to perform useful network administration tasks. To achieve this we have defined a set of visually orthogonal metaphors that we map onto network metrics. These metrics can be taken from various network monitoring and network intrusion detection systems.

We suggest the use of 3D game engines as a platform for building these visualisations as much of the hard work of building a 3D application has been removed. We also gain

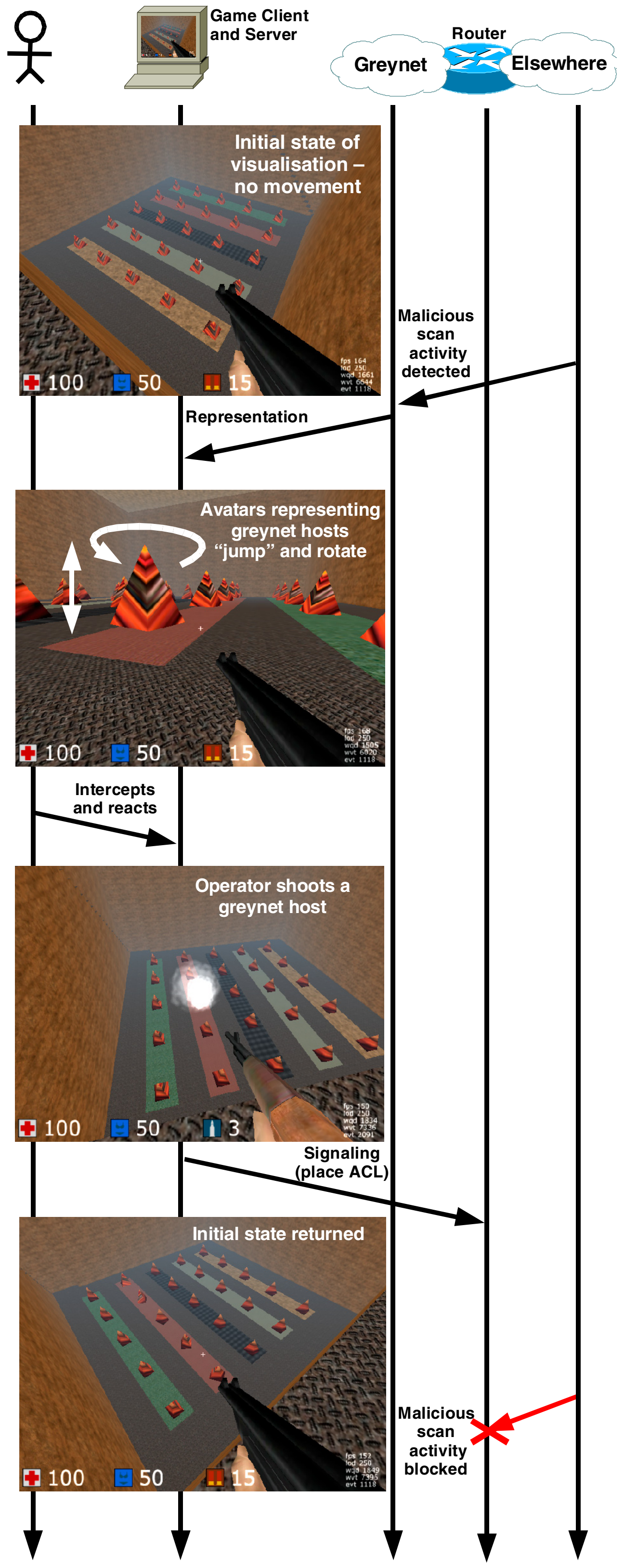
Game Client and Server
Router
Greynet
Elsewhere
Initial state of visualisation – no movement
Malicious scan activity detected
Representation
Avatars representing greynet hosts "jump" and rotate
Intercepts and reacts
Operator shoots a greynet host
Signaling (place ACL)
Initial state returned
Malicious scan activity blocked

Figure 5: A day in the life of a fully immersed network operator - single user case

from the other features already resident in most 3D game engines, such as sophisticated visual appearance, multi-player support (to support collaboration within the same visualised network) and optimised code (to ensure efficient real-time graphics rendering).

We have defined an initial set of interaction metaphors that are easy to implement within the context of existing 3D game-engines. Currently we envisage network operators might shoot (with gun), heal (with syringe) and tweak (with pliers). These allow a user to interact with the virtual environment and instigate changes in real time back onto the running network. 'Shooting' places firewall access control lists, using 'pliers' places bandwidth restrictions and 'heal' undoes these operations. Future implementations may choose different interaction metaphors - this is limited primarily by the underlying game-engine and the preferences of the target users. We have implemented a large number of our defined features in our prototype software that leverages the 3D game engine Cube for implementation. We can successfully run a test network where network scans are detected by a greynet. This greynet state is then represented in the 3D world by avatars spinning and "jumping" to visually alert the network operators to a network anomaly. The operators, using their superior pattern recognition skills, can then detect and shoot the alerting avatars to instantiate a firewall access control list on a cisco router, preventing any further scans.

We have closed with our future research direction, to continue prototype advancement and perform human factors research along with a call for more standard methods of darknet/greynet state inspection by creating a MIB extension allowing interaction via the SNMP protocol.

10. ACKNOWLEDGMENTS

Thanks to Carl Javier for creating the 3D pyramid avatars used in this paper's prototype software.

This paper has been made possible in part by a grant from the Cisco University Research Program Fund at Community Foundation Silicon Valley.

11. REFERENCES

[1] Bro. http://bro-ids.org/, Aug 2006.
[2] Brutal file manager. http://www.forchheimer.se/bfm/, Aug 2006.
[3] Cube (game/3d engine). http://cube.sourceforge.net/, Aug 2006.
[4] The freebsd project. http://www.freebsd.org/, Aug 2006.
[5] Honeyd honeypot project. http://www.honeyd.org/, Aug 2006.
[6] HP openview management software. http://www.managementsoftware.hp.com/, Aug 2006.
[7] id software, doom 1, 2, quake 1, 2 and 3. http://www.idsoftware.com/, Aug 2006.
[8] Microsoft directx: Home page. http://www.microsoft.com/windows/directx/default.aspx, Aug 2006.
[9] Mrtg: The multi router traffic grapher. http://people.ee.ethz.ch/ oetiker/webtools/mrtg/, Aug 2006.
[10] Nagios. http://www.nagios.org/, Aug 2006.
[11] Netflow v9. http://www.cisco.com/en/US/products/ps6601/ products_white_paper09186a00801341b2.shtml, Aug 2006.
[12] Nmap security scanner. http://www.insecure.org/nmap/, Aug 2006.
[13] opengl. http://www.opengl.org/, Aug 2006.
[14] Rrdtool. http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/, Aug 2006.
[15] Snort. http://www.snort.org/, Aug 2006.
[16] Sun microsystems project looking glass. http://www.sun.com/software/looking_glass/, Aug 2006.
[17] The team cymru darknet project. http://www.cymru.com/Darknet/, Aug 2006.
[18] Valve software. http://half-life2.com/, Aug 2006.
[19] Visualizing internet topology at a macroscopic scale. http://www.caida.org/analysis/topology/as_core_network/, Aug 2006.
[20] P. Abel, P. Gros, C. Santos, D. Loisel, and Paris. Automatic construction of dynamic 3d metaphoric worlds: An application to network management. In *Visual Data Exploration and Analysis VII, volume 3960*, pages 312–323, Jan 2002.
[21] M. Bailey, E. Cooke, T. Battles, and D. McPherson. Tracking global threats with the internet motion sensor. Technical report, October 2004.
[22] R. Ball, G. A. Fink, and C. North. Home-centric visualization of network traffic for security administration. In *VizSEC/DMSEC '04: Proc. of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 55–64, New York, NY, USA, 2004. ACM Press.
[23] P. Biddle, P. England, M. Peinado, and B. Willman. The darknet and the future of content distribution. In *In Proc. of the 2002 ACM Workshop on Digital Rights Management.* ACM Press, 2002.
[24] J. Case, M. Fedor, M. Schoffstall, and J. Davin. Simple Network Management Protocol (SNMP). RFC 1157 (Historic), May 1990.
[25] D. Chao. Doom as an interface for process management. In *CHI '01: Proc. of the SIGCHI conference on Human factors in computing systems*, pages 152–157, New York, NY, USA, 2001. ACM Press.
[26] B. Cheswick, H. Burch, and S. Branigan. Mapping and visualizing the internet. In *USENIX Annual Technical Conference, General Track*, pages 1–12, 2000.
[27] B. Claise. Ipfix protocol specification, Internet Draft, June 2006.
[28] G. Conti and K. Abdullah. Passive visual fingerprinting of network attack tools. In *VizSEC/DMSEC '04: Proc. of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 45–54, New York, NY, USA, 2004. ACM Press.
[29] E. Cooke, M. Bailey, Z. M. Mao, D. Watson, F. Jahanian, and D. McPherson. Toward understanding distributed blackhole placement. In *WORM '04: Proc. of the 2004 ACM workshop on Rapid malcode*, pages 54–64, New York, NY, USA, 2004. ACM Press.

[30] B. Feinstein, G. Matthews, and J. White. The intrusion detection exchange protocol (idxp), Internet-Draft, Oct. 2002.

[31] P. Fröst, M. Johansson, and P. Warrén. A computer game virtual environment for collaboration. In *GROUP '01: Proc. of the 2001 International ACM SIGGROUP Conference on Supporting Group Work*, pages 1–2, New York, NY, USA, 2001. ACM Press.

[32] J. R. Goodall, W. G. Lutters, and A. Komlodi. I know my network: collaboration and expertise in intrusion detection. In *CSCW '04: Proc. of the 2004 ACM conference on Computer supported cooperative work*, New York, NY, USA, 2004. ACM Press.

[33] J. R. Goodall, W. G. Lutters, P. Rheingans, and A. Komlodi. Preserving the big picture: Visual network traffic analysis with tn. In *VIZSEC '05: Proc. of the IEEE Workshops on Visualization for Computer Security*, page 6, Washington, DC, USA, 2005. IEEE Computer Society.

[34] W. Harrop and G. Armitage. Intuitive real-time network monitoring using visually orthogonal 3d metaphors. In *Australian Telecommunications Networks & Applications Conference 2004 (ATNAC2004)*, December 2004.

[35] W. Harrop and G. Armitage. Defining and evaluating greynets (sparse darknets). In *LCN '05: Proc. of the The IEEE Conference on Local Computer Networks 30th Anniversary*, pages 344–350, Washington, DC, USA, 2005. IEEE Computer Society.

[36] W. Harrop and G. Armitage. Greynets: a definition and evaluation of sparsely populated darknets. In *MineNet '05: Proc. of the 2005 ACM SIGCOMM workshop on Mining network data*, pages 171–172, New York, NY, USA, 2005. ACM Press.

[37] A. Komlodi, P. Rheingans, U. Ayachit, J. R. Goodall, and A. Joshi. A user-centered look at glyph-based security visualization. In *VIZSEC '05: Proc. of the IEEE Workshops on Visualization for Computer Security*, page 3, Washington, DC, USA, 2005. IEEE Computer Society.

[38] B. Kot, B. Wuensche, J. Grundy, and J. Hosking. Information visualisation utilising 3d computer game engines case study: a source code comprehension tool. In *CHINZ '05: Proc. of the 6th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction*, pages 53–60, New York, NY, USA, 2005. ACM Press.

[39] K. Lakkaraju, W. Yurcik, and A. J. Lee. Nvisionip: netflow visualizations of system state for security situational awareness. In *VizSEC/DMSEC '04: Proc. of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 65–72, New York, NY, USA, 2004. ACM Press.

[40] S. Lau. The spinning cube of potential doom. *Commun. ACM*, 47(6):25–26, 2004.

[41] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage. Inferring internet denial-of-service activity. *ACM Trans. Comput. Syst.*, 24(2), 2006.

[42] D. Moore, C. Shannon, G. M. Voelkery, and S. Savagey. Network telescopes: Technical report. Technical report, April 2004.

[43] A. Oline and D. Reiners. Exploring three-dimensional visualization for intrusion detection. In *VIZSEC '05: Proc. of the IEEE Workshops on Visualization for Computer Security*, page 14, Washington, DC, USA, 2005. IEEE Computer Society.

[44] C. Papadopoulos, C. Kyriakakis, A. Sawchuk, and X. He. Cyberseer: 3d audio-visual immersion for network security and management. In *VizSEC/DMSEC '04: Proc. of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 90–98, New York, NY, USA, 2004. ACM Press.

[45] E. M. Raybourn and N. Bos. Design and evaluation challenges of serious games. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 2049–2050, New York, NY, USA, 2005. ACM Press.

[46] T.-M. Rhyne. Computer games and scientific visualization. *Commun. ACM*, 45(7):40–44, 2002.

[47] C. Scott, K. Nyarko, T. Capers, and J. Ladeji-Osias. Network intrusion visualization with niva, an intrusion detection visual and haptic analyzer. *Information Visualization*, 2(2):82–94, 2003.

[48] H. Takemura and F. Kishino. Cooperative work environment using virtual workspace. In *CSCW '92: Proc. of the 1992 ACM conference on Computer-supported cooperative work*, pages 226–232, New York, NY, USA, 1992. ACM Press.

[49] Y. Waern and D. Pargman. Design and use of muds for serious purposes (workshop session)(abstract only). In *CSCW '96: Proc. of the 1996 ACM conference on Computer supported cooperative work*, page 2, New York, NY, USA, 1996. ACM Press.

[50] K. Wakita and K. Shimamura. Smartcolor: disambiguation framework for the colorblind. In *Assets '05: Proc. of the 7th international ACM SIGACCESS conference on Computers and accessibility*, pages 158–165, New York, NY, USA, 2005. ACM Press.