Re-Rezz BBS #3

Bigger is Better

You're connected to a datafortress, scouring the financial archives of a small corporation, hoping to find something juicy for a bit of extortion. But there's nothing—just terminals, door controls, and camera links. "What's wrong?" you mutter as you patrol the virtual space within the fortress.

Suddenly, a new data wall rises before you, its security clearance higher than before. As you crack it, the virtual sound of an alarm blares in your ears. "My program didn't trigger it, I'm sure," you think frantically as you scramble to breach the wall and reach the other side.

"One LDL?" is the only thought you have time for before diving into it. No time for wondering while the bloodhounds are closing in.

Sometimes you want to do quick runs, but other times you want to dive deep and create a big, intricate operation.

A datafortress can be as complex and fun as a puzzle, and for that, you go bigger.

The Basics

The bread and butter of a netrunner's job is stealing data from a datafortress. But what kind of data are they after, and how do they find it?

Information is stored in databases, folders, and files within the system. These are typically represented as memory blocks containing local data. However, some data is stored on the "cloud," an external system usually maintained by another corporation.

To access external data, you must enter a protected LDL icon (using a standard Code Gate opener), which will transport you to another datafortress with its own set of security measures.

Keep in mind that cloud services pride themselves on their security, so be prepared for extreme countermeasures if they detect your unauthorized breach.

What's in a Typical Cloud Memory Unit (CMU)? : Mostly mundane stuff. Work databases, operations documentation, personal info, provider/client databases, personnel calendars, and the like.

What's in a Local Memory Unit (MU)? : The same as a CMU if the datafortress doesn't use cloud services, plus the usual corporate detritus: PowerPoint presentations, internal messages, local info, images, and the juicy bits. The really sensitive, top-secret data that a corporation wouldn't dare trust to an external service—that's the treasure you're after.

Bigger is Better

The previous schematic is for a single shard server—the kind you'd find in small enterprises, lone buildings, or similar setups.

Single shard? you might be asking. Well, when things scale up, a single server can't handle everything. Nor is it desirable. If your factory machines aren't working as expected, you don't want to reboot security cameras, doors, or international operations.

So, you connect multiple servers (called shards) and divide the workload among them. Now, if your factory is having problems with the machinery, you don't need to unplug the security systems.

And yes, every shard has its own defenses.

That means more work for you. Aren't you happy?

What Does This Mean?

It means you now have to defeat various data walls to reach your desired destination.

You first break into the login/net server, which only contains minimal operational MUs, then search for a shard index, hidden LDL or other internal data walls protecting those exit points ... and try to breach its security.

Jumping from one point of the net into another and repeating your steps in and endless loop of pain and Black ICE.

A World of Shards

In summary ...

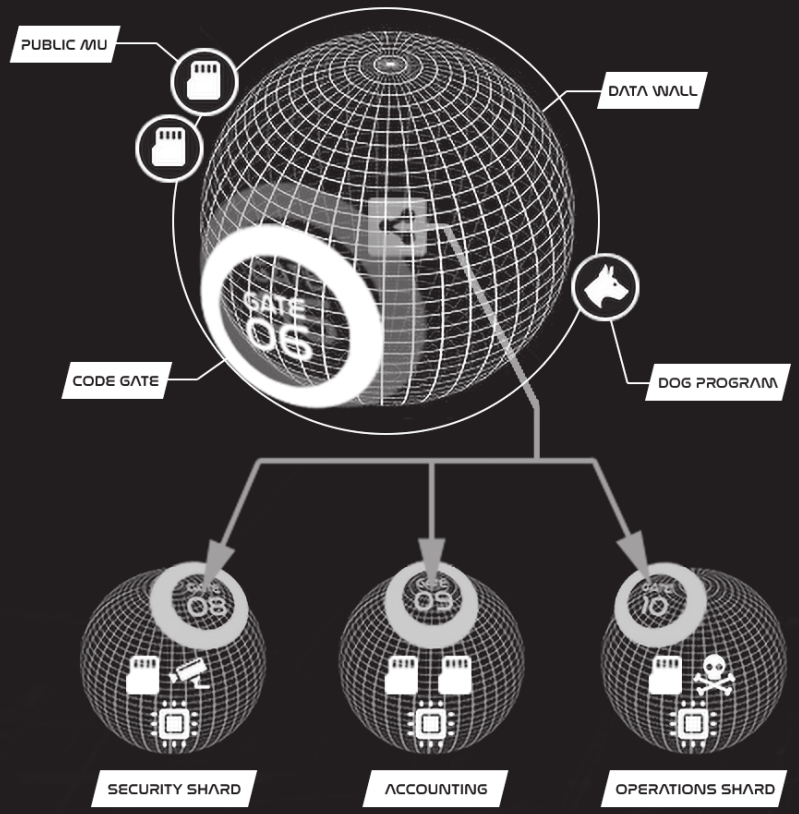
Each datafortress is divided based on its functions—security, robot control, accounting, and so on. You need to find the LDL that takes you to these shards.

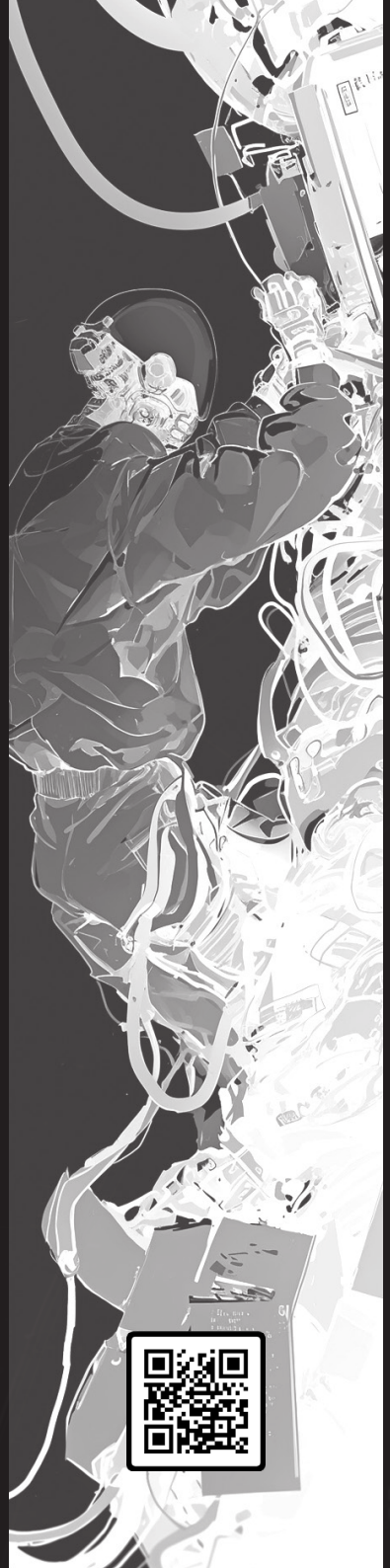
Each shard has its own Code Gate and Data Walls, which you'll need to breach. You can only access them from the LDL inside the main datafortress.

Moreover, each shard likely has its own sysop or, worse, an AI.

Additionally, some shards are only accessible once you've connected to a specific shard. For example, if you want access to the black ops file, you'll need to hack into the initial datafortress (Login), then into the Management shard, and finally into the Internal Security shard.

Since each shard is different, the corporation may decide the first two shards aren't critical and have a wafflerunner controlling them. But the last one? Well, choom, that one will have a top-tier operator running state-of-the-art decks loaded with the preemest black ICE.





Who Runs It?

A lot of datafortresses have enough CPU power to be managed by AI.

But you won't find many sites actually using AI for management. Why? Because AIs are predictable, reprogrammable, and exploitable. They don't learn or react like humans (and you really don't want an AGI wandering freely), and they don't just spawn spontaneously—you have to pay someone to create them.

Don't get me wrong, AIs are terrifying and can flatline you faster than your deck can render a warning. But when someone figures out how to exploit an AI, the news spreads like wildfire among netrunner BBS.

Soon enough, a bunch of exploit programs are being copied and traded.

At that point, those AIs are as secure as wet toilet paper.

It's easier, cheaper, and more reliable to use human sysops to monitor the system. Sure, they can be bargained with, extorted, or even bribed... but don't count on it. After all, they are sell-outs working for corporations; not many will trade a well-paid, secure job for a meager bribe. And let's be honest, if you had the kind of money to secure that sysop's future, you wouldn't be doing this.

As an extra, if the sysop screws up and ends up with their brain fried, the corporation doesn't have to pay their salary. It's a win-win situation for those coldblooded freaks.

If a sysop is working locally, they'll use the mainframe as their terminal to administer the datafortress. This means multiple actions and blazingly fast programs. It also means you know exactly where they are in the building, so someone could "unplug" them in the real world.

Don't believe me? Check Rache Bartmoss' Guide to the Net, page 144, "Netrunning from Mainframes."

If not, they'll use their own cyberdecks and work like a netrunner would.

And that's the gist of it.

Try not to get yourself flatlined next time you assault a multi-shard datafortress.