Who Wrote Sobig?

Version 1.0:	19-August-2003.
Version 1.1:	25-August-2003.
Version 1.2:	19-November-2003.
Version 1.3:	17-July-2004. This sanitized variation for public release. Scheduled for release: 1-November-2004.

Table of Contents

1 About This Document

August 18, 2003 was a day of infamy in the world of computer software malware. The Sobig virus, as it was affectionately named by its the anti-virus industry, infected hundreds of thousands of computers within just a few short hours. W32.Sobig.F@mm was a mass-mailing, network-aware worm that sent itself to all the email addresses it could find, worldwide.

Within two days after Sobig was released, an estimated $50 million in damages were reported in the US alone. China had reported over 30% of email traffic had been infected by Sobig, equivalent to over 20 million users! After interrupting freight operations and grounding Air Canada, Sobig went on to cripple computing operations within even the most advanced technology companies, such as Lockheed Martin. Sobig was so virulent that on November 5, 2003 Microsoft, in coordination with the FBI, Secret Service, and Interpol, setup the Anti-Virus Reward Program. Backed by $5 million from Microsoft, the program offered a $250,000 bounty for information leading to the arrest and conviction of the Sobig author.[1]

As the one year anniversary of the Anti-Virus Reward Program bounty for Sobig approaches, we felt this was an appropriate time to publicly release the current state of our Sobig forensic investigation. Appropriately, the authors of this document have chosen to release it anonymously for many reasons, some of which are:

- By releasing the information publicly, we hope to increase tips to law enforcement concerning the Sobig authorship and spur efforts toward apprehension of the malware author(s);
- This document shows how computer forensics can identify virus authors. The computer forensic methods demonstrated throughout this document have been utilized to successfully identify authors of other viruses as well;
- Our focus is the objective analysis of Sobig. It is our contention, position, and belief that associating this paper with any specific company, organization, group, or individual will only serve to detract from the investigation.

The following public PGP key is provided for document validation, with the private key component safely locked away as to eliminate any future chance of a lost key pair. Any individual or entity that claims authorship should be able to validate their ‘authorship’ by signing a message with the corresponding PGP private key.

- The included PGP public key prevents unscrupulous people from claiming ownership of this document or attempting to collect the Microsoft bounty;
- As this document is present on multiple mirrored sites and has been turned over to law enforcement, anyone modifying the PGP public key will be unable to pass a fake key for potential bounty award;
- This PGP public key will only be included is this document. Other documents, where malcontents attempt to place our ownership on other findings, should be considered forgeries unless they include a message signed with the PGP private key.

In the event that any individual or entity may be able to identify the authors of this document, we urge you to respect our request for anonymity.

[1] Ironically, our investigation into the identification of the likely Sobig author(s) and corresponding findings had already been concluded and passed on to law enforcement over two months prior to the Microsoft bounty offer. The bounty was not our incentive.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP

mQGiBEF5UyURBACZ9jsrBaowh4npI0ac84RVD9fzzR9oW9x4TkfPNlddzzR4uhko
CsKUK9IfeEz7C6a6pMKbwB83xjBedhjRE5zMfqns4kSY33XJirGzRHWXWbD7n1zn
WFfq84VBCezDmUy6cy4+zipQDyPYs+u8YfCMUfS2XCHXwD8lqg5jvIzsiwCglpkp
z4+x1qvUrVNIkJlxeYzbDOsD/3Y/DHvNmNa8LGKTyF7KFgD5TTBSnPW5tpzdXkcp
+UVg09A6qHagx90LGOA6l0P8oOJYPGaCvuQV7Br2zqqAw5PUouwxFPJTsWrmU25N
lQ83jf8WaJTvU46/TXg8H+ctKeCLa4R6h4+dDkRjxSeKMuGbK8yZAVOv+VkANRci
sFumA/sHb/+ovnRRp0rQ6xSz0HDOzJNNCF3ghZ4M+h13o7Ll4FCYVjr2pWeoeWKH
80KZ8QWXfHMv4UmKekJ/ghI0AbUN/2ZLSqMTcQqZjqZdGivHPNG1rSk0qPupyQi+
+Cp7UM49r8kfC7eMmwqzd3VIix1+HC+nlxmkk6E/rYXaDKvzOLQ4QXV0aG9ycyAo
QXV0aG9ycyBvZiAiV2hvIFdyb3RlIFNvYmlnPyIpIDxhdXRob3JzQGRvbWFpbj6I
XwQTEQIAHwUCQXlTJQUJA8JnAAQLBwMCAxUCAwMWAgECHgECF4AACgkQ68sDbPEA
3GOKQwCcC2Lg98QoRbDhwvm4y2KK0i6gTYsAoIcFHzo1owPyi3uObA80V+wnr11O
uQINBEF5U2MQCACH2+h+bY7cMe6ti87JRkx50Ian0B4g8m3ugvNN0qXd2YNdFOuD
igrMi/SOLLb8tuGGOTEVkkNwtkejnP0Nc8tppxbJkEvG+MKxbOScSUEHswNAjTG0
qdMTT3HsGQZbgoxEE5RzW5ZqMizUTRvLAWB98LCZWeyfzkX14iflJw9m3RtsAFc0
TSV/WLsPH+o+ula8hkdc7klEEDRguLMkcOSttgwa4UbDota81pZ1poEbYRDGUFMT
VMTrkFNV4mCRsUPsDkOn9spdlBAkC6U/h8kcfWdNT7V8V6mdW7kV4WmJ24xGro7d
RxHMiGVVYTMesaW8chz2xO5rG+nwS4yyX723AAMFB/9EMSUKaucuBmFnKwxGHMAb
D01m8AAIE4chZF4fhhF9d8lnSMqyxwm/X9uSvUGD/3hF2QwvMzQVTFiy5ChYsIwz
TuK2WyQMbgWkmvlUn2RNUP68Il+6ewWyupTZdbSHSxiurWyyAqX9gBLkmX4ndOeG
e2LPwBZ+GWLK8KFGGZTyanLkLfZ3DUwj9eMXaW5EYhoEQ/lIcp7m3kJ5k3WlVALA
MMU5WJRbpY+7Q8Nn+zS0d59p2HALIEUvXxlueHpoxvhYPKo4j3xzB5/8CE3RdP8y
UeNpLSVieR7CDFSYuGvHwsKi+zi0ihHFyjosj9CEJ1rsKAEIlu6P3LdpYKcGEFDM
iEwEGBECAAwFAkF5U2MFCQPCZwAACgkQ68sDbPEA3GMMAACeJoVCaAIw6NZA78CH
MQpkb1m8nDUAn2UT+n7BsfFUPK3hvtfCCvf9ywP5
=7hlJ
-----END PGP PUBLIC KEY BLOCK-----
```

2 Overview

Sobig was a virus specifically designed to aid the anonymity of spammers. Sobig opened up services that enabled spammers to relay their emails anonymously. Although publicly the motivation and author of the Sobig virus is unknown, through the use of forensics and profiling, we have identified a very likely suspect and motive.

Our research indicates that Ruslan Ibragimov of Moscow, Russia, and/or Ibragimov's development team, authored the Sobig virus. Ibragimov himself is the author of Send-Safe, a bulk mailing tool product that was explicitly designed for sending unsolicited email (spam).

Our investigation will demonstrate:

- **Advanced knowledge**: Ibragimov has demonstrated an advanced knowledge of Sobig outbreaks.
    - The releases of Send-Safe coincide with Sobig releases;
    - New features in Send-Safe coincide with Sobig features;
    - A specific spam group that use Send-Safe was observed relaying through Sobig-infected systems as much as two weeks before the official outbreak.
        - This same group has been observed using specific versions of Send-Safe prior to public release (using pre-released software);
        - The time that the group was observed using Sobig (prior to public announcement) corresponds with the Internet Storm Center recording an increase in port scans for Sobig-infected systems.

- **Necessary skills**: Based on the attributes and overall functionality of Sobig, the developer would require the following skills:
    - Knowledge of Microsoft Visual C++;
    - Self-compressing executables;
    - Email and spam;
    - Proxies.

  Ibragimov has demonstrated these skills and knowledge: Send-Safe is a spam tool designed to send email using proxies, written in Microsoft Visual C++, and self-compressed.

- **Source code access**: Sobig and Send-Safe share a common source code base.
    - Unique source code creates unique opcodes within the executable code.
        - Both the Sobig and Send-Safe software share large sections of common opcode sequences, implying the same source code;
        - The common source code is used to both generate and send email;
    - The email headers are unique to Send-Safe;
    - Sobig includes code for an email header that it does not use.
        - This unused code appears in the same order as the Send-Safe executable – and the code is used in Send-Safe;
    - Ibragimov has not publicly released the source code to Send-Safe (or any of his other programs, to our knowledge).
        - Send-Safe predates Sobig by a few years, indicating Ibragimov had access to the original code base used to develop Sobig;
    - Ibragimov has demonstrated a pattern of reusing source code.
        - Large blocks of opcodes found in Send-Safe appear in other programs created by Ibragimov.

- **Plausible motive**: As Send-Safe provides a list of open proxies to subscribers, there is a clear financial motive for Ibragimov to have created the Sobig worm.
    - As Sobig opens additional ports, this provides more open proxies for Send-Safe subscribers.

Based on these items that have been identified, we contend that Ruslan Ibragimov, or Ibragimov's development team, authored Sobig in order to support and extend the Send-Safe customer base.

3 Spam and Virus Release History

Sobig appears designed specifically to assist spammers with anonymity. Sobig opened new network services that enabled spammers to relay their emails. So the first part of this investigation will begin to take a look at the spam groups.

To effectively send anonymous spam, there are three items required:

1. A bulk mailing tool;
2. An individual or group to operate the tool;
3. A set of anonymous proxies for relaying the message(s).

3.1 Identifying Tools

The generation of email spam to a potential list containing millions of email addresses must be done quickly and efficiently. To accomplish this within a reasonable timeframe, an automated bulk-mailing tool is typically used. Bulk-mailing tools can usually be identified by the unique attributes and residues that are placed within the generated email's header. These particular header fields, their ordering and value patterns within the email are generally consistent for any bulk-mailing tool, but differ between each email. Identification of email by a particular spam tool can be accomplished by simply grouping the spam messages based on their attributes. Within the scope of this document and investigation, the bulk-mailing tool is named **Send-Safe** (www.send-safe.com). Depending on the specific version of Send-Safe, each has a slightly different header format (Fig.1).

Fig. 1 Sample Send-Safe headers. Different revisions have minor header differences.

Send-Safe 2.0	Send-Safe 2.14
Reply-To: <fgtwy@aol.com> Message-ID: <022a36a14e7c$7882b4e6$5be88aa1@ueoqrf> From: <fgtwy@aol.com> To: select Subject: Home Based Business Date: Wed, 22 May 0102 23:13:14 -0600 MiME-Version: 1.0 Content-Type: multipart/mixed; boundary="---- =_NextPart_000_00A2_87B13A0B.B6003A33" X-Priority: 3 (Normal) X-MSMail-Priority: Normal X-Mailer: Microsoft Outlook, Build 10.0.2616 Importance: Normal	Message-ID: <1d5c002d720c$8de7722e$9213aa50@wfksiy.iud> From: <yankee-eril@yahoo.com> To: <***user1@domain***> Cc: <***user2@domain***>, Subject: You'll be HUGE! In just 2 weeks! Date: Fri, 06 Jun 2003 05:15:57 +0100 MIME-Version: 1.0 Content-Type: multipart/alternative; boundary="---- =_NextPart_9D8_B8E1_9212F5FC.091B2CAD" X-Priority: 3 X-MSMail-Priority: Normal X-Mailer: ardeli79352a789

Send-Safe is the only known bulk-mailing tool that generates these email headers in this specific order and with these value ranges. Even though there are over 14 revisions of the Send-Safe software, most of the versions 2.0 thru 2.14 have very similar layouts. Subtle differences shown in the above examples include the spelling of "MiME-Version" versus "MIME-Version", the correction of a Y2K defect (0102 should have been 2002), an addition of a period in the Message-ID hostname, and the dropping of the uncommon "(Normal)" comment in the X-Priority field.

3.2 Identifying Individuals and Specific Groups

Although the publicly available Send-Safe software is a common bulk-mailing tool, not all Send-Safe users employ the software the same way. Based on specific user-configurable functions, Send-Safe users can usually be identified[2] by the unique macros employed, their values, sending habits, contents and the basic functionality selected. Users of a specific feature set are considered to be in the same 'spam gang'.

Known Send-Safe 'spam gangs' include the DHS Club, "Mr. Yahoo", and the Send-Safe Spam Group (SSSG).

[2] Identifying an individual does not necessarily imply knowing their name. It usually implies that all emails with identical traits are likely from the same unknown person.

The DHS Club can be easily recognized by their content, as they appear to operate a pyramid scheme out of Florida.[3] "Mr. Yahoo" is an individual that always uses a forged sender address to offer porn, while claiming to be from "yahoo.com", and uses a Message-ID that always ends in a capital letter sequence (Fig. 2).

Fig. 2 Send-Safe header from Mr. Yahoo

```
Message-ID: <674701c38b6f$27fdba53$3e2fcf90@BCD>
From: "gabriela 21939" <bigcfmjmfhnkl@yahoo.com>
To: "jessa" <user@domain>
Subject: leemav Find Your Dream Fuckhole With Drastically Simple Criteria Searches!
Date: Sun, 5 Oct 2003 18:33:12 +0000
MIME-Version: 1.0
Content-Type: multipart/alternative;
         boundary="----=_NextPart_000_54F9_01C38B90.0174D88E"
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express 6.00.2800.1158
X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2800.1165
X-UID: 605
```

SSSG stands out from the fray as a unique spam group or 'gang'. The SSSG operating method includes:

- **Rotating software**. Most spam groups do not change their software until it becomes ineffective. Conversely, SSSG appears to rotate their software regularly. Since everyone in the group does not rotate their software at the same time, a graph of their software revisions over time appears as multiple, overlapping bell curves (Fig. 3).
- **Rotating sending methods**. The majority of spam groups stick with specific sending methods, such as the use of open web or socks proxies. SSSG stands out because they rotate their sending methods. Initially, they were observed sending directly without any proxies, but then moved to open proxies; initially ones that they operated, and then later to open proxies provided by Sobig.
- **Pre-release software**. SSSG has been observed using new releases of Send-Safe software prior to the actual public release of the software. In many cases, SSSG appears to have used previously un-released versions of Send-Safe, which appear to correspond with jumps in the public release Send-Safe software revisions.

[3] Source: "The Distributed Home Spamming DHS Club", <http://www.theclubbuiltonspam.com/>, 22-October-2003.

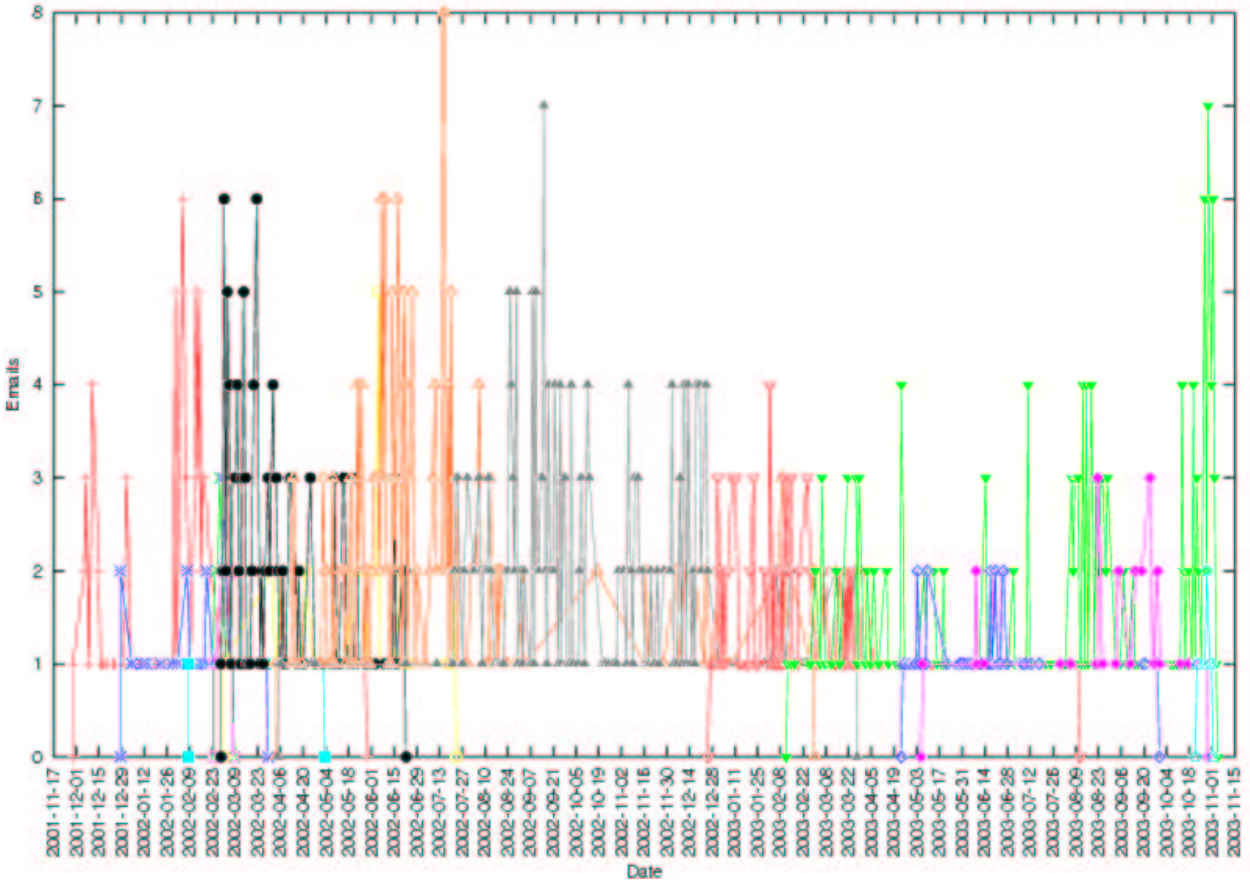


Fig. 3 Software revisions from SSSG rotate over time. Every color indicates a different software revision.

3.3 Identifying Open Proxies and Usage

Open proxies are the third item that an anonymous spam group requires. Open proxies allow a sender to relay messages while keeping their true IP address anonymous. If the sending system's IP address happens to be tracked to the spam, the tracker will find a system with an open proxy server, masking the true sender's IP address. Multiple proxies are used for three main reasons:

1. Heavy use of a single proxy may be noticed and cause the proxy to be disabled;
2. Some proxies place a limit on the number of times an individual may relay through it;
3. The bandwidth of the proxy server is unknown, so by relaying through many proxies, the spam sender can increase their throughput.

Send-Safe provides four methods for sending spam:

1. **Use your own**. The user can provide a list of proxies, leaving it up to the user to acquire the list;
2. **Send-Safe proxy scanner**. Send-Safe provides a proxy scanner for use with the Send-Safe bulk-mailing tool;
    - As of scanner version 1.5, the unregistered scanner will send all discovered proxies to Send-Safe for use with the registered bulk-mailing tool; if you find a proxy using the unregistered proxy scanner, then other Send-Safe users also gets to use the proxy.
3. **Send-Safe proxy list**. Send-Safe can provide a list of proxies at an additional cost;
4. **Send direct**. If no proxies are available, the Send-Safe bulk-mailing tool can send email directly. The user may also select an option to send directly.

3.4 Conjecture: Send-Safe and Sobig

The Send-Safe software appears to have direct links to the Sobig virus.

Multiple releases of Send-Safe software correlate with releases of Sobig, both in chronological order and functionality (Fig. 4). Each revision of the Sobig virus had opened a set of network services that can be used to relay spam email; Sobig was likely created to open proxy servers for use by spam senders.[4]

Fig. 4 Correlations between Send-Safe and Sobig releases.

Software	Release Date	Notes
Send-Safe 2.0	09/2002	Send-Safe 2.0 introduced encrypted proxy lists. This happened after a newsgroup acquired the proxy and address list (35Meg file), and made it public.
Send-Safe 2.1	11/20/2003	2.1 introduced SSL to the server, along with encrypted proxy data. The same newsgroup called it "trivial" to decode, decoded it in under a week, and posted the information.
Sobig-A	1/09/2003	No expiration. Executable packaged with tElock. Believed to be a proof-of-concept test.[5] Opens proxies on ports 555, 608, and 1180-1185.
Send-Safe 2.09	3/19/2003	Compiled on March 18, 2003. (Possible prior releases between 2.1 and 2.09, but we did not track until March.) Change in registration: Minor registration charge for using your own proxies. Major registration charge if Send-Safe provides proxies.
Send-Safe 2.13	5/18/2003	Compiled on May 18, 2003. **Supports more proxy types**. (HTTP, HTTPS, Socks, etc.)
Sobig-B	5/18/2003	Compiled on May 16, 2003. Deactivates May 31, 2003. Executable packaged with UPX. Believe to be the first non-proof-of-concept release. Release correlates with Send-Safe 2.13 release. Provides **many types of proxies**, most (all?) are supported by Send-Safe 2.13, but not supported by Send-Safe 2.09.
Send-Safe 2.14	5/30/2003	Compiled on May 30, 2003. New registration system. All proxies found by Send-Safe proxy scanner are now sent to a central Send-Safe server for all users. **More complex encryption system** for transferring the proxy list.
Sobig-C	5/31/2003	Compiled on May 30, 2003. Deactivates June 8, 2003. Executable packaged with UPX. **More complex encryption system**. Release correlates with Send-Safe 2.14 release.
Sobig-D	6/18/2003	Deactivates July 2, 2003. Believed to be a premature release or a test due to limited spread and overlapping expiration date with Sobig-E. Sobig-D used the same ports as the previous Sobig releases.
SSSG	6/18/2003	SSSG observed relaying through a system with ports **2280-2285** open.
Sobig-E	6/25/2003	Compiled on June 24, 2003. Deactivates July 14, 2003. Executable packaged with tElock. Proxy ports changed to 1555, 2001, and **2280-2285**.
Send-Safe 2.14	6/28/2003	Web page updated. Re-release of the same code; no code change.
SSSG	7/1/2003	SSSG is observed using a new, unreleased version of Send-Safe. This version is believed to be a pre-release of Send-Safe 2.16, possibly 2.15.
SSSG	8/9/2003	SSSG observed relaying through a system with ports **3380-3385** open. The open services match Sobig-F.
Sobig-F	8/18/2003	Compiled August 17, 2003, public announcements 2 days later. Deactivates September 10, 2003. Executable packaged with tElock. Includes encrypted command structure and remote control. The proxy ports changed to 2555, 3001, and **3380-3385**.
SSSG	9/30/2003	SSSG is observed using a new, unreleased version of Send-Safe. This version is later determined to be the final release of Send-Safe 2.16, before the public release.
Proxy Scanner 1.6	9/30/2003	New Send-Safe Proxy Scanner released. Coincides with a sudden end of SSSG traffic.
Send-Safe 2.16	10/6/2003	The official release of Send-Safe 2.16. Coincides with a sudden restart of SSSG traffic.
Send-Safe 2.17a	10/17/2003	Compiled on October 17, 2003. The web page still indicates version 2.16.
HPH 1.0	11/11/2003	The first release of Send-Safe's "Honeypot Hunter".
Send-Safe 2.17b	11/18/2003	The web page was updated to "2.17 beta" but the web page's date still says October 6, 2003.

[4] Source: "Sobig.a and the spam you receive today." by Joe Stewart, Lurhq. <http://www.lurhq.com/sobig.html>.
[5] Source: "Evolution of the Worm." by Joe Stewart, Lurhq. <http://www.lurhq.com/sobig-e.html>.

4 ISC Malware Scans and Public Announcements

At present, it is inconclusive whether the developers of Send-Safe actually created the virus or simply sponsored the creation. It is equally ambiguous whether SSSG sponsored the creation of Sobig, requested additional proxies, or just utilized what was originally provided to them. What we do know is that SSSG and Send-Safe were observed to be using Sobig-infected systems before the actual public announcement.

4.1 Public Announcements

An email was received from SSSG on June 18, 2003. This email was received from IP address "200.75.209.87", located in Cable Onda, Panama (City), Panama (Country). A scan of that particular host revealed the following:

2285/tcp = SMTP server
2284/tcp = POP3 server
2283/tcp = WinGate FTP server
2281/tcp = telnet server of some kind. Prompt says "MNGTR>"
2001/tcp = unknown server.

Why so many open systems services were running was inexplicable at the time. It simply appeared that more than likely the system had been compromised. Between the 18th and 25th of June 2003, additional email messages originating from SSSG were observed coming from other systems with the identical open ports.

Concurrently, the previously mentioned Send-Safe groups DHS and "Mr. Yahoo" were explicitly relaying through both open socks servers and HTTP proxies – systems that did not have the same open ports as the hosts used by the SSSG. Additionally, there were no other non-Send-Safe spam groups observed originating from this particular type of compromised host system.

The Sobig-E virus was officially identified and publicly announced on June 25th, 2003. Sobig-E opens the identical services and ports used by the SSSG. Simultaneously, the Internet Storm Center had begun reporting an increase in scans for port 2280 just prior to the Sobig-E public announcement. This increase in port 2280 scans directly corresponded with the SSSG's employment of infected systems.

An email received from SSSG on August 9th, 2003 displayed a similar Sobig-F pattern. Using the 'nmap' and 'nessus' utilities to scan the host system, we identified a telnet service with an "MNGTR>" prompt and many open services on ports 3380-3385. Additionally, on August 16th, 2003, the Internet Storm Center began noticing an increase in port 3380 scans. The Sobig-F malware that opens these particular ports and this particular "MNGTR>" service was not publicly announced until August 19th, 2003 (Fig. 5).

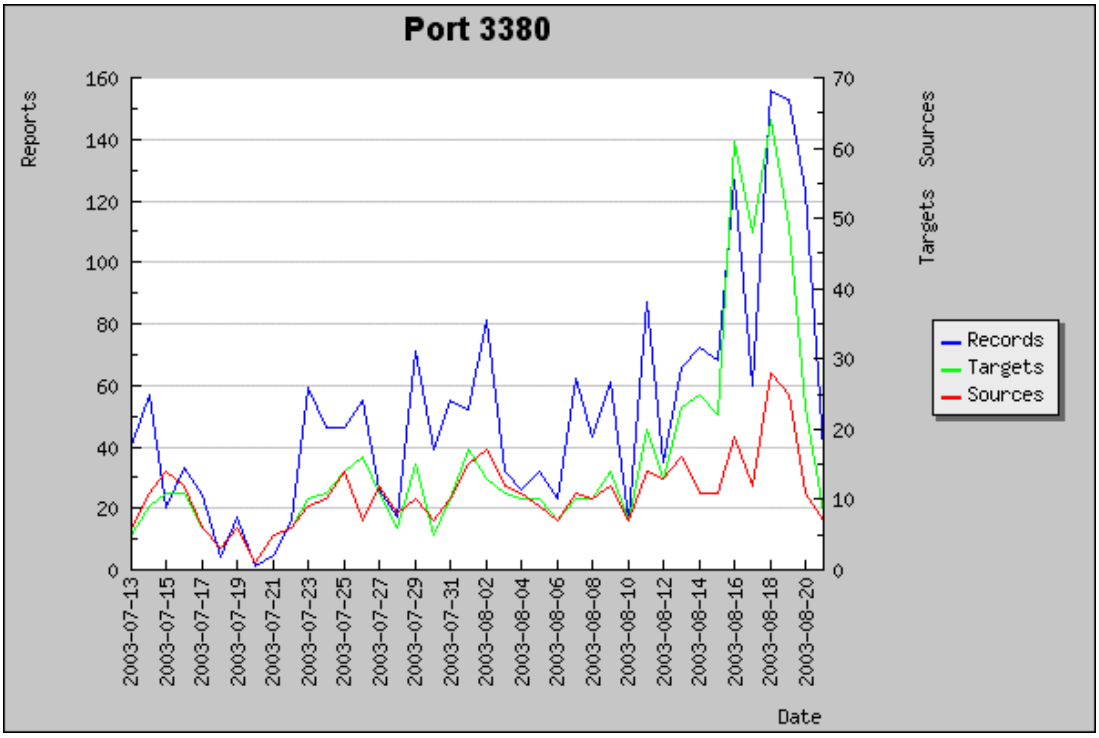


Fig. 5 The Internet Storm Center had recorded an increase in port 3380 scans a few days prior to the August 19, 2003 public announcement of the Sobig-F malware.

Soon after the Sobig-E and Sobig-F viruses had been publicly announced, other spam groups began to relay through open proxies via Sobig-infested systems. There was only a time span of a few days to a few weeks between the public announcement and the observed port use by other parasitic spam gangs. Of the identified Send-Safe spam gangs, only a few had not modified their scanning to include the Sobig proxy ports.

Although the Sobig releases appear to have direct links to the SSSG, Sobig was not the only malware being employed by the SSSG prior to the Sobig public announcements. SSSG had been observed relaying via systems with an open web proxy on other ports, specifically 5490/tcp as early as June 9th, 2003 (Fig. 6). This directly coincides with an increase in 5490 port scans and actually predates the use of other spam groups relaying through systems on this specific open port. However, the actual service that was responsible remains unidentified. It is also unclear if SSSG had actually discovered the proxy on their own, if Send-Safe had discovered it and notified SSSG, or if some other custom malware was directly responsible.

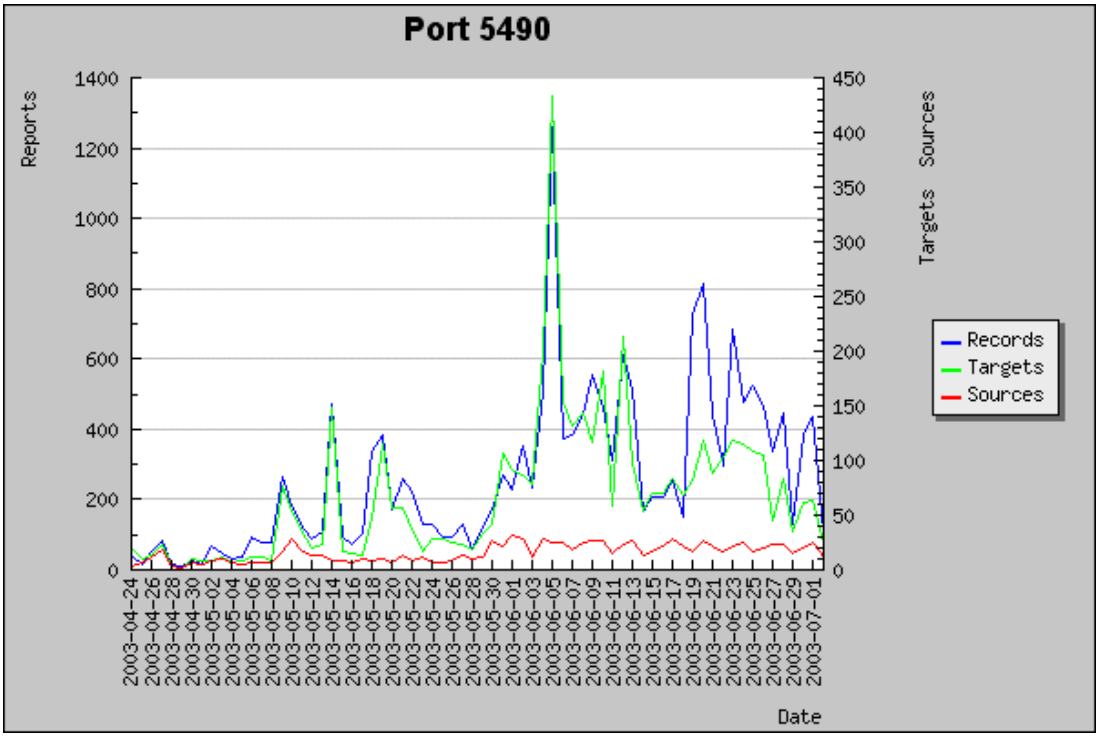


Fig. 6 The Internet Storm Center reported an increase in scans for port 5490 during a time when SSSG was observed using systems with port 5490 open.

5 So, Who Wrote Sobig?

Since a public confession has not been made as to the original Sobig malware author(s), we have identified four particularly unique attributes that must be known in an attempt to identify the author(s).

1. Advanced knowledge of the virus;
2. Skill to develop it;
3. Access to the source code;
4. Motive to write the malware.

5.1 Advanced Knowledge of Sobig

During our investigation it became apparent that some individuals appeared to have advanced knowledge of the Sobig release. Typically anyone with advanced knowledge would either be in direct/indirect contact with the malware author(s), or have the unique potential to be the actual malware author(s). Even though anyone with a malware software toolkit could have created Sobig, virus writers for the most part rarely retain the information within themselves, at least for a long time.

This investigation documents three very substantial correlations indicating potential knowledge of the actual malware author:

1. SSSG had been using Sobig prior to the public announcement. This indicates insider knowledge between SSSG and the Sobig virus. This also implies a direct/indirect relationship between the Sobig virus author(s) and SSSG;
2. SSSG had been using Send-Safe versions prior to public release. This implies special treatment and a possible direct relationship: SSSG knows the Send-Safe team, and the Send-Safe team knows SSSG;
3. The Send-Safe bulk-mailing tool software releases directly coincide with the Sobig releases. This also implies a close relationship between the Send-Safe software and the Sobig malware author(s).

However, it is unclear if the Sobig author(s) is an actual member of the Send-Safe development team, a part of the SSSG or an independent. Due to the apparent relationship between SSSG, Send-Safe and Sobig, it is very plausible that any of these participants could lead to the Sobig author(s).

As SSSG appears to be a sizable organization, it would seem unlikely that any individual within the group would actually know the Sobig author(s). It is also unlikely that a specific individual from within SSSG can be easily identified due to their use of Send-Safe.

Contrastingly, a Send-Safe developer is known who operates the Send-Safe product line (application, bulk-mailer, proxy scanner): Ruslan Ibragimov. Although it is unknown if Ibragimov is the sole developer, or if his team co-authored Sobig, it seems highly plausible that Ibragimov may actually know the Sobig author(s).

5.1.1 About Ruslan Ibragimov

Ruslan Ibragimov is a known software developer who has created a variety of tools:

- Log Analyzer[6] for the IMail server (imla.exe), written in Delphi;
- IcqVampire[7], a tool for harvesting email addresses from the ICQ white pages;
- Send-Safe, a bulk-mailing tool, compiled using C++.

Through public forum communications, we have determined Ibragimov to possess the following technical skills:

- Programming languages: Delphi, Microsoft Visual C++, Pascal;
- Databases: SQL, Oracle, Sybase;
- Applications: SMTP, Proxy clients, ICQ, and Windows GUI development.

[6] http://www.ipswitch.com/support/IMail/helperapps.html
[7] http://www.happywebonline.it/archivio/10-2002/speart.htm

Ibragimov's address on record is 12 Krasnokazarmennaya, 111250, Moscow, Russia. Previous known email addresses (no longer active) include <rusoil@mail.ru>[8], <ruslan@rusoil.net>[9], and <ruslan@albea.rb.ru>[10]. As the latter two email addresses are from universities, it is probable that Ibragimov may have attended Ufa State Aviation Technical University around 1998, and the Ufa State Petroleum Technical University sometime in 2000 and 2001.

5.1.2 About Send-Safe

Although an individual may have authored Send-Safe, it is a fairly large and complex application that requires detailed knowledge of networking, multi-threaded programming, SSL, cryptography and more. Although Ruslan Ibragimov demonstrates skill and knowledge in these areas, there is no definable identifier that he is the sole developer. In our review of the Send-Safe executable code, there are numerous traits that suggest multiple developers, which indicates Ibragimov may actually manage a team of developers.

5.2 Skill to Develop Sobig

Developing the Sobig malware would require a thorough knowledge of Microsoft Visual C++, self-compressed executables, email and spam, and proxies. Since Sobig is believed to have been released by posting the virus directly to newsgroups, knowledge of newsgroup (NNTP) would also be required. Unlike the Send-Safe executable, Sobig is very small in size, was created with special care to hide the author's identity, and could have easily been developed by a single individual.

Unless otherwise specific, the following versions of Sobig and Send-Safe that are listed below refer to Sobig-F and Send-Safe 2.16.

5.2.1 Skill: Microsoft Visual C++

The Sobig virus was compiled using Microsoft Visual C++, which is the same type of compiler used to compile Send-Safe. Although it is unclear if the exact same Microsoft Visual C++ compiler was used, Send-Safe appears to also have been compiled using Microsoft Visual C++ and Borland C++.

5.2.2 Skill: Self-Compressed Executables

The phase-1 Sobig malware executable was "packed", i.e. a self-compressed executable. The Sobig-A, Sobig-E, and Sobig-F were packed using tElock by tHE EGOiSTE/TMG; while Sobig-B and Sobig-C were packed using UPX. Both tElock and UPX packing tools include compression and encryption and very commonly used among virus writers. These tools are freely available and require very little skill to use.

Similar to tElock and UPX, the Send-Safe executable was compressed using ASPack by Alexey Solodovnikov. , Even though the packing software itself is different, this demonstrates that the Send-Safe developers have an awareness of self-compressed and encoded executables.

5.2.3 Skill: Email and Spam

Well designed email messages are one method used by Sobig to self-propagate and use spam-techniques to hide the origination system. Two techniques that Sobig employs when propagating by email are "direct delivery" and "bounced delivery". Direct delivery is when the email is sent directly to the recipient. In this approach, the header information is forged, including the originating hostname and "From:" email address. Key items from the Sobig email header are:

- The header order:
  - From, To, Subject, Date, Importance, X-Mailer, X-MSMail-Priority, X-Priority, MIME-Version, and Content-Type;
- A "Message-ID" header is not present;
- The "Date" contains a programming error.

[8] Source: WHOIS domain registry information for the domain "send-safe.com".
[9] Source: newsgroup postings circa 2000-2001.
[10] Source: newsgroup postings circa 1998.

  - If the random time zone is negative, then it is displayed with two negative signs. For example, "`Date: Fri, 4 Jul 2003 0:08:45 --0700`";
- The "X-Mailer" value is "Microsoft Outlook Express 6.00.2600.0000".

The bounced delivery propagation technique is when email is sent to a non-existing email address, and a mail daemon returns the undelivered email back to the sender. By forging a recipient's email address within the "From" address, the mail daemon then completes the delivery by including the malware in the "undeliverable" email reply.

Authoring Sobig would require knowledge of email generation, delivery, and spam-techniques for obscuring the originating host. Ruslan Ibragimov has demonstrated these skills with the Send-Safe bulk-mailing tool.

5.2.4 Skill: Proxies

To remain anonymous, a spam senders must relay through an open proxy. Many malware applications create their own proxy servers for spam to relay. For example, W32.Mimail.gen@mm worm[11] contains code for opening a proxy server. In contrast, Sobig contains no code for opening proxy servers; Sobig retrieves the proxy tool "WinGate" and installs it on the infected system.

As the skill necessary for installing WinGate is minimal, it is likely that the Sobig author(s) has limited or no experience at developing proxy servers. Ruslan Ibragimov has not been observed demonstrating a proficiency in proxy server development.

5.2.5 Skill: Newsgroups

It is widely believed that the outbreak of each Sobig virus began when infected files were placed within newsgroup forums (NNTP). As newsgroups are extremely common and easy to post messages to, it is likely that the virus author is familiar with newsgroups and is not someone new to NNTP. Although there may be many individuals and groups that fit this particular profile, it cannot be used to rule-out Ruslan Ibragimov; Ibragimov has been posting to newsgroups since at least 1998.

5.3 Access to Sobig Source Code

There are many similarities between the Sobig and Send-Safe executables, including the generated email header formats, programmer coding conventions, and identical opcode sequences.

5.3.1 Similar: Header Ordering

Different spam tools will generate different email headers. The email header that was generated by Sobig is very similar to the header generated by Send-Safe.

Send-Safe 2.14 email header	Sobig-F email header
Message-ID: <036a55d03c4c$8127a7d8$6bc21ae1@hhutma> From: <alastairbaileymf787@flash.net> To: <***user@domain***> Subject: **You're ~Approved~* Date: Wed, 26 Mar 2003 13:21:50 -0000 MIME-Version: 1.0 Content-Type: multipart/mixed; boundary="----=_NextPart_000_00B0_40A21C5D.D8133A18" X-Priority: 3 (Normal) X-MSMail-Priority: Normal X-Mailer: QUALCOMM Windows Eudora Version 5.1 `Importance: Normal`	From: <info@nofear.ch> To: <***user@domain***> Subject: Thank you! Date: Tue, 26 Aug 2003 14:29:47 --0400 X-MailScanner: Found to be clean Importance: Normal X-Mailer: Microsoft Outlook Express 6.00.2600.0000 X-MSMail-Priority: Normal X-Priority: 3 (Normal) MIME-Version: 1.0 Content-Type: multipart/mixed; boundary="_NextPart_000_0661EBB9"

Both Sobig and Send-Safe email headers contain the same four generic-email "required" header fields; From, To, Subject, and Date at the beginning and in the same order. But, Sobig does not contain a Message-ID field.

[11] Source: http://hq.mcafeeasap.com/dispVirus.asp?virus_k=100796

Following the required headers are the "optional" headers for generic email traffic. Even though these fields are considered optional, they are always present in email generated by both Send-Safe and Sobig. The optional headers are listed in the exact opposite order (Importance, X-Mailer, X-MSMail-Priority, and X-Priority). The value of these options vary with different Send-Safe versions, but the values used in Sobig-F match the values from Send-Safe 2.14.

Lastly, the MIME-Version and Content-Type headers are listed in the exact same order, including the same indentions.

Although these subtle differences suggest separate source code, the similarities suggest that Send-Safe was the template, and not other mailing programs such as Outlook, Netscape, The Bat!, or AMS.

As these other independent email tools generate their headers with very different ordering, it would seem unlikely that the Sobig author(s) determined the email headers and values independently.

5.3.2 Similar: Coding Conventions

The Sobig malware has a significant number of coding conventions that are very similar to Send-Safe:

- **String concatenations**. The Sobig virus uses static email header values, i.e. the string "X-Mailer: Microsoft Outlook Express 6.00.2600.0000" does not vary. Although the string is static, the code that generates it concatenates two independent strings: "X-Mailer: " and "Microsoft Outlook Express 6.00.2600.0000". The concatenation suggests that the developer reused code with a static value ("Microsoft...") rather than simply encoding the entire static string.

  The majority of bulk mailing tools concatenate the field and value pairs like Sobig. Send-Safe uses a concatenation function to combine the header information with the dynamic value strings. This suggests that Sobig used existing code for generating an email header, and then modified it to concatenate static strings rather than simply hard-coding the static header values.

- **Use of "%s"**. When concatenating strings, there are many different approaches that can be used, i.e. a C++ programmer can use "." or "+" for concatenation, or the old-style C notation "%s". There are also string concatenation functions such as 'strcat()'.
  - When generating the email header, both Send-Safe and Sobig do not use "%s" to concatenate strings. But, when establishing a connection to an email server (SMTP connection), both Send-Safe and Sobig use "%s". The strings used by Send-Safe and Sobig are identical. For example, "RCPT TO: <%s>\r\n\0" (hex: "52 43 50 54 20 54 4F 3A 20 3C 25 73 3E 0D 0A 00").

  The use of multiple concatenation approaches, with identical approaches for identical functionality is a significant correlation.

- **Unused strings**. The Sobig executable contains the string "Message-ID", even though the virus never generates a Message-ID in email that it sends. This particular unused string also suggests reuse of code.

- **String ordering**. In compiled executables, strings are generally listed in the order that they appear within the source code and are not necessarily listed in the order that they are used.[12] The string ordering in Sobig is similar to the ordering in Send-Safe. For example:

[12] Executables are created in a two-step process. First, the source code is compiled into an object file. Second, all the object files are linked into a single executable. All of the strings in the source code are grouped in the object file. The order of the strings in the object file match the order they were found in the source code (from top of file to bottom). The linker groups all strings from the object files in the same area of the executable. The order of the strings matches the object file linking order. Large groups of strings may be in a different order depending on the linking, but the local cluster of strings will not change as the executable is created.

Sobig-F strings	Send-Safe 2.16 strings
_NextPart_001_%8.8X This is a multipart message in MIME format %s: %s Message-ID MIME-Version	----=_NextPart_FFF_FFFF_FFFFFFFF.FFFFFFFF Message-ID: Microsoft Outlook Reply-To: {%From%} From: {%From%} {%To%} Subject: {%Subj%} MIME-Version: 1.0

Since Send-Safe is a much more complex program than Sobig, Send-Safe has many more strings to indicate the additional options. However, the overall ordering, with the unused "Message-ID" string following the "_NextPart" string and preceding the "MIME-Version", indicates a semi-unique function ordering (unique order of strings in the source code) between the two code bases.

Even though both programs generate email and are expected to use these strings, neither program lists these strings in usage order. (For usage ordering, the Message-ID would be listed first, followed by From, To, Subject, etc., and ending with MIME-Version, Content-Type, and _NextPart boundary. The usage order should match the email header ordering.) This particular unique string ordering suggests source code with functions that use these strings being listed in the same order. The matching string order indicates a similar (or same) code base. It seems very possible that Send-Safe and Sobig used the same basic code base, while the Sobig author(s) simply removed unnecessary functionality from the source code.

The following unlikely coincidences suggest a common-code base or the same author(s) between Sobig and the Send-Safe code base:

1. Use of general string concatenations when creating the email header, but "%s" when communicating to an SMTP server;
2. Similar string ordering;
3. Sobig's unused string ("Message-ID") listed in the same string order (between "_NextPart..." and "MIME-Version") as Send-Safe.

There are no similarities to other email programs when comparing the Sobig and Send-Safe string ordering. The string order comparison was done against the Atomic Mail Sender (AMS) version 2.26, The Bat! version 1.62, Microsoft's Outlook Express for Windows 2000 and Outlook 2000[13], elm version 2.5 PL6, mutt version 1.2.5.1i, and pine version 4.44. Therefore, these other email programs are very unlikely to share any code base with Sobig or Send-Safe.

5.3.3 Similar: Opcode Sequences

When creating executable software, source code is written and then compiled into object code. Object code is then linked into operation code (opcode) sequences. High-level programming languages such as Visual C++, provide many different ways to create the same functionality. Different implementations for programming the same functionality will result in different opcode sequences. Knowing this, there are only a few conditions that can generate identical opcode sequences:

1. The compiler may optimize small sections of simple functionality (such as a loop to initialize a variable). The optimization may result in identical opcode sequences from different coding implementations. The identical opcode sequences are unlikely to be larger than 64 bytes;
2. The compiler may have generic templates for common functionality, which is usually the case for the start of any executable program, calls to common library functions, and padding between linked objects;
3. Programs that include library functions ("statically linked libraries") will contain the same opcode sequences due to the inclusion of identical code.

[13] Based on string order comparisons, only Microsoft's Outlook and Outlook Express are similar. These programs very likely share a common code base.

During this investigation, we compared Sobig-F with the Atomic Mail Sender (AMS) version 2.26, Sobig-E, Send-Safe 2.17a installer, and Send-Safe 2.17a installed executable[14]. The comparisons were done using a full-sequence comparison system (similar to the Unix program 'diff', but it compares all sections of code with all other sections – see Appendix A for the detailed comparison).

Here were some of our findings:

- **Comparing Sobig with AMS**. AMS and Sobig contain common high-level functionality, as both programs generate and send email. Although there are many ways to create this functionality in source code, it is extremely unlikely that two people working independently would generate similar opcode sequences for this type of functionality. From the results of our comparisons, the first 1K of memory indicated that they are very similar types of executables. The padding at the end of memory boundaries match, indication similar compilers. However, no part of Sobig's executable opcode sequence matches AMS. A lack of similarity from this comparison is expected since, although both programs send email, different developers are believed to have created them.

- **Comparing Sobig-F with Sobig-E**. Unlike the AMS comparison, there are very large blocks of similar and identical opcode sequences between these different revisions, indicating a very similar code base. There are some differing segments, indicating source code changes between Sobig-E and Sobig-F. The similarity between these two programs is expected since the same developer is believed to have created both programs; Sobig-F is a later revision of Sobig-E.

- **Comparing Sobig with the Send-Safe installer**. The Send-Safe installer is an executable archive that installs Send-Safe. The only common high-level functionality between Sobig and the installer is the creation of files on the local computer. Like the AMS comparison, there are similar sections with the initial 1K of memory and end of memory boundaries, and there is some similarity with library function calls and linking with static libraries, indicating similar compilers. However, the complete lack of matching executable opcode sequences clearly indicates completely different code bases. This lack of similarity is expected since the developer of the installer-building program was unrelated to the Sobig or Send-Safe developers.

- **Comparing Sobig with Send-Safe executable**. The Send-Safe executable has the same high-level relationship with Sobig as AMS since they both generate and send email. Yet there are significant blocks of similar and identical opcode sequences between Sobig and Send-Safe. These similar, and in very many instances, identical sections of opcode sequences, are not due to compiler optimizations or common library functionality, as some of these similar blocks are over 1K in size. The similar code appears to focus on the generation and sending of email. These similar/identical opcode sequences imply the same source code base.

Within the scope of our comparison we can conclude that the developer of the Sobig virus reused source code from Send-Safe for the creation of code that generates and delivers email in Sobig[15]. The source code to Send-Safe is not public or open sourced, as Ruslan Ibragimov owns the source code and appears to keep the code private.

In summary for any non-technical readers, this significant appearance of similar and identical opcode sequences within the Send-Safe and Sobig software should be considered as significant as finding a fingerprint on a murder weapon. Although this very close similarity does not constitute guilt, it suggests a very strong correlation.

---

[14] The Send-Safe installed executable is encrypted and compressed. When it is executed, the program unpacks itself in memory. For the comparison, we executed Send-Safe 2.17a and then stored all the program memory (25 Megabytes) into a file. The file contains both the unencrypted and uncompressed Send-Safe executable and all dynamically loaded libraries. Our test was performed on a clean-install system that had never accessed Sobig, ensuring that any similarities are not due to Sobig residing on the system.

[15] Send-Safe predates Sobig by more than a year. It is unlikely that Ruslan Ibragimov acquired the source code from Sobig for use in Send-Safe.

5.4 Motive to Write Sobig

Senders of spam typically relay their email messages through open proxy servers in a continuing effort to obscure the true sending host. With the proliferation of blacklists and other anti-spam systems, spam senders are finding it more and more difficult to locate available open proxy servers. By opening multiple proxy services on millions of compromised systems, a spam sender could very quickly and anonymously relay messages without the fear of being identified.

Sobig provides the following two benefits for spam senders:

1. Sobig opens multiple proxy servers on systems that are not blacklisted;
2. Sobig spreads very quickly, infecting and re-infecting millions of systems in under a week.

These benefits provide spam senders with a very large base of open proxy servers. Even though most of the infected systems will be cleaned within a week, there will be some systems that will remain infected to continually provide open proxies for weeks or even months.

We believe that Sobig was most likely written to support spam software. Any user or developer of spam mailing software, including Ruslan Ibragimov and Send-Safe, would be financially eager to leverage malware such as Sobig.

6 Conclusion

Based on our findings and the items we have identified within the scope of this document, we strongly believe that Ruslan Ibragimov of Russia, and/or Ibragimov's development team, authored Sobig. Ibragimov has demonstrated an advanced knowledge of Sobig outbreaks, the skills necessary for developing Sobig, access to the source code base used to develop Sobig, and a clear motive for creating the Sobig malware.

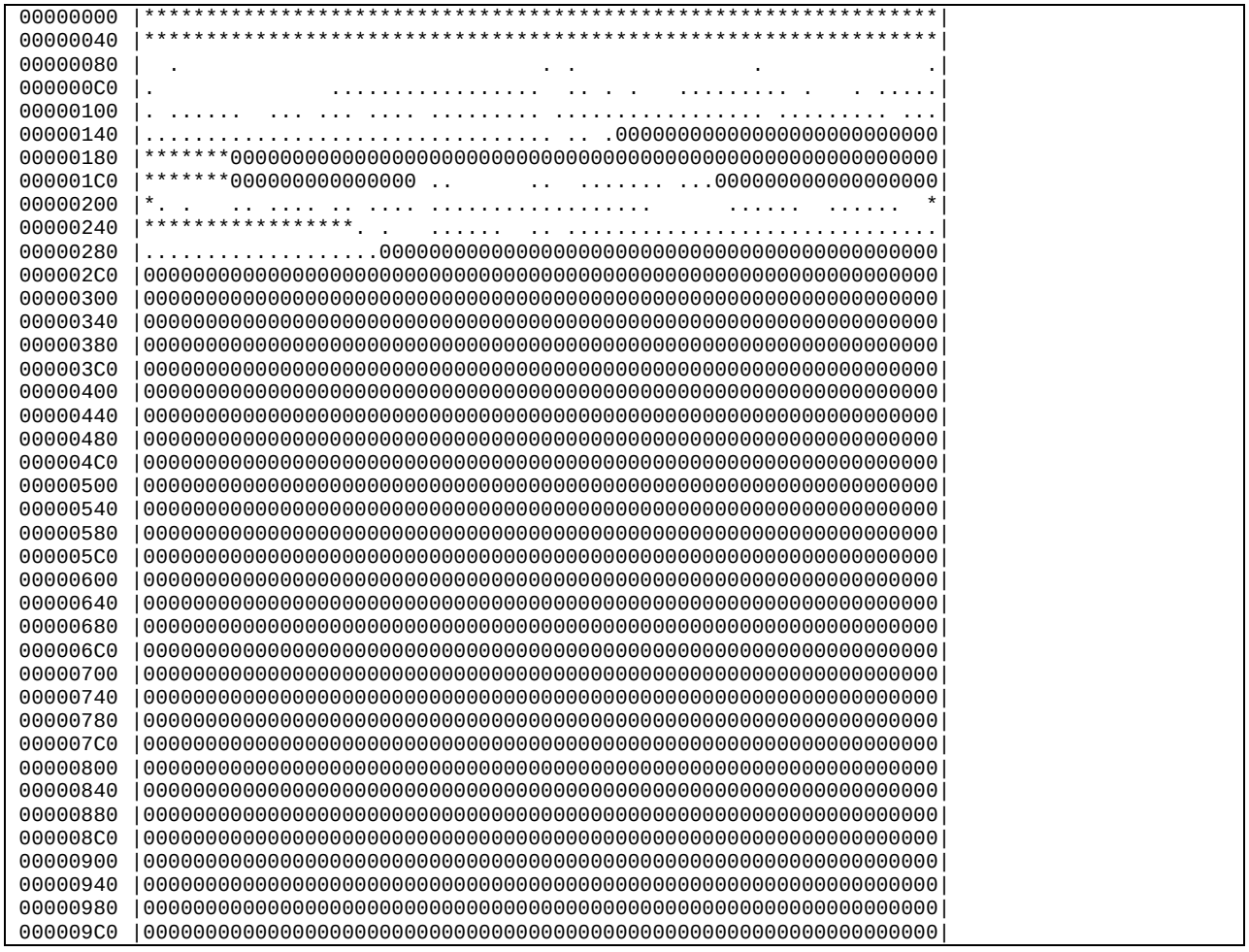
7 Appendix A

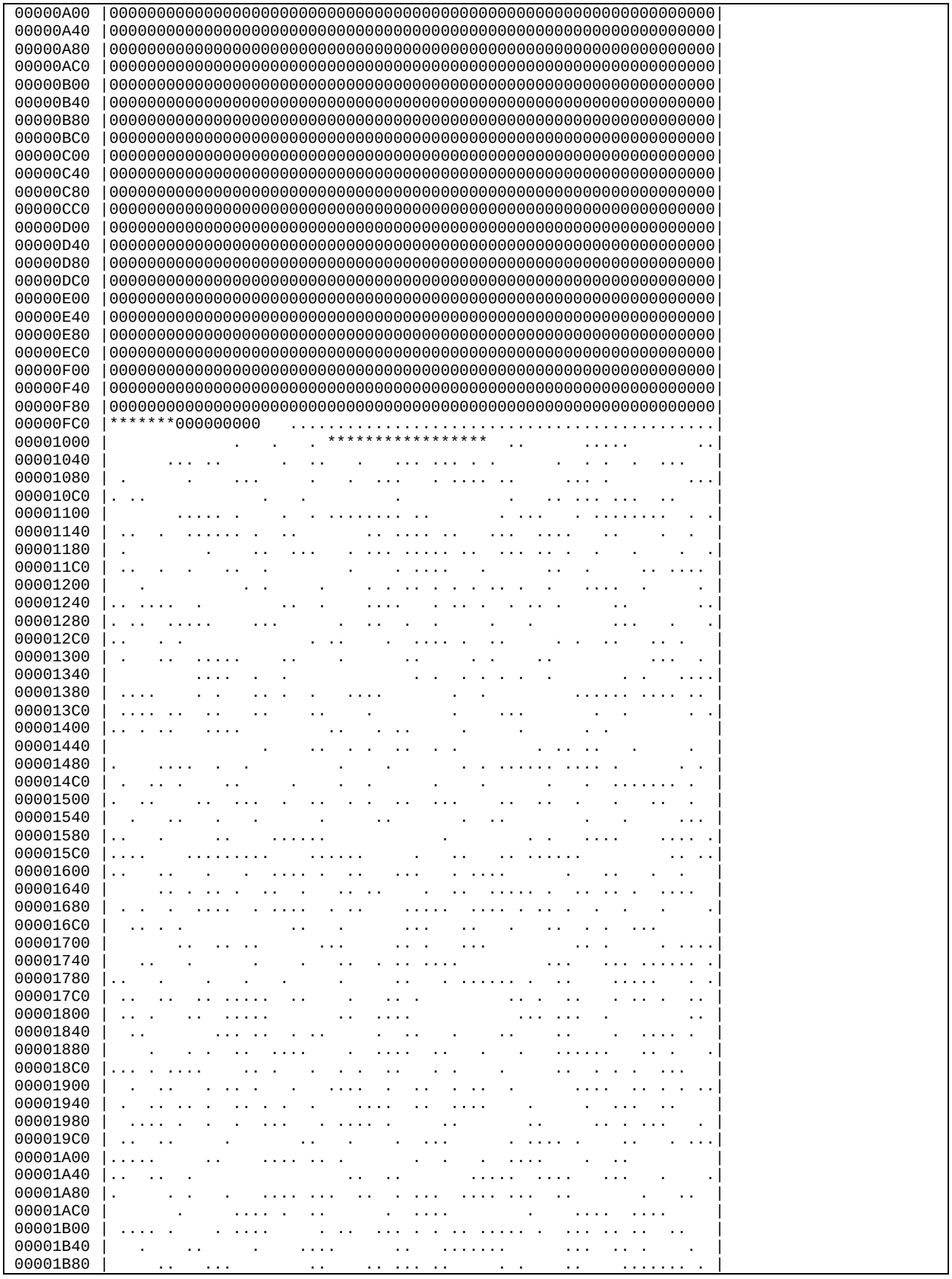
The following table shows a comparison of the bytes in the Sobig-F executable when compared with the unpacked Send-Safe 2.16 executable.

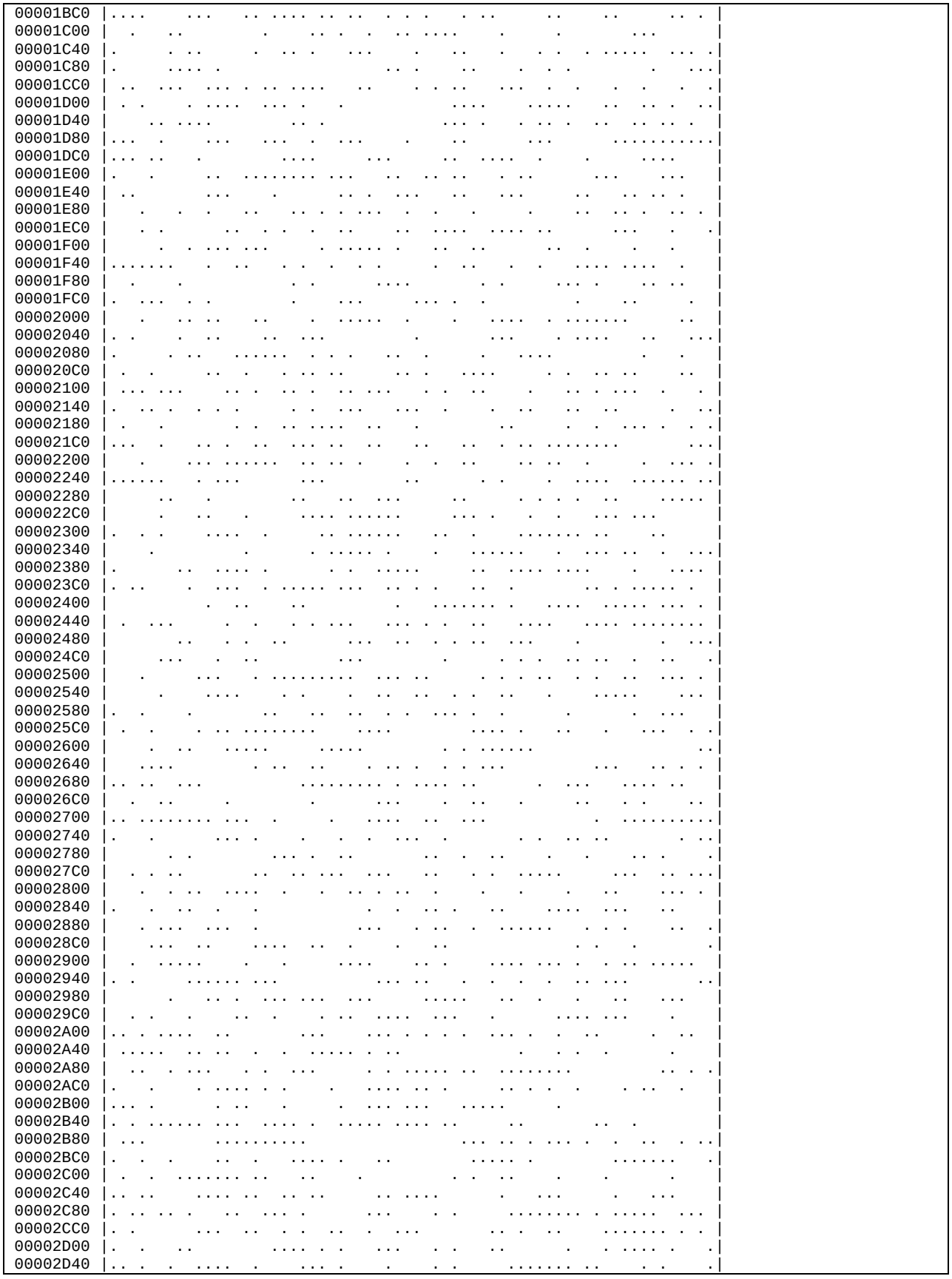
- The memory address refers to the location within the Sobig-F executable;
- A space (“ ”) indicates a byte in Sobig that has no corresponding match in Send-Safe. Only absolute matches (exact same byte value) are permitted;
- A dot (“.”) indicates a byte value in Sobig that may correspond with a sequence in Send-Safe;
- A star (“*”) indicates a byte value in Sobig that positively corresponds with a sequence in Send-Safe. A positive match assumes at least 16 sequential matching bytes, showing that the byte is the same and in the same context;
- A zero (“0”) indicates a large sequence of zeros, usually used for padding between memory sections.

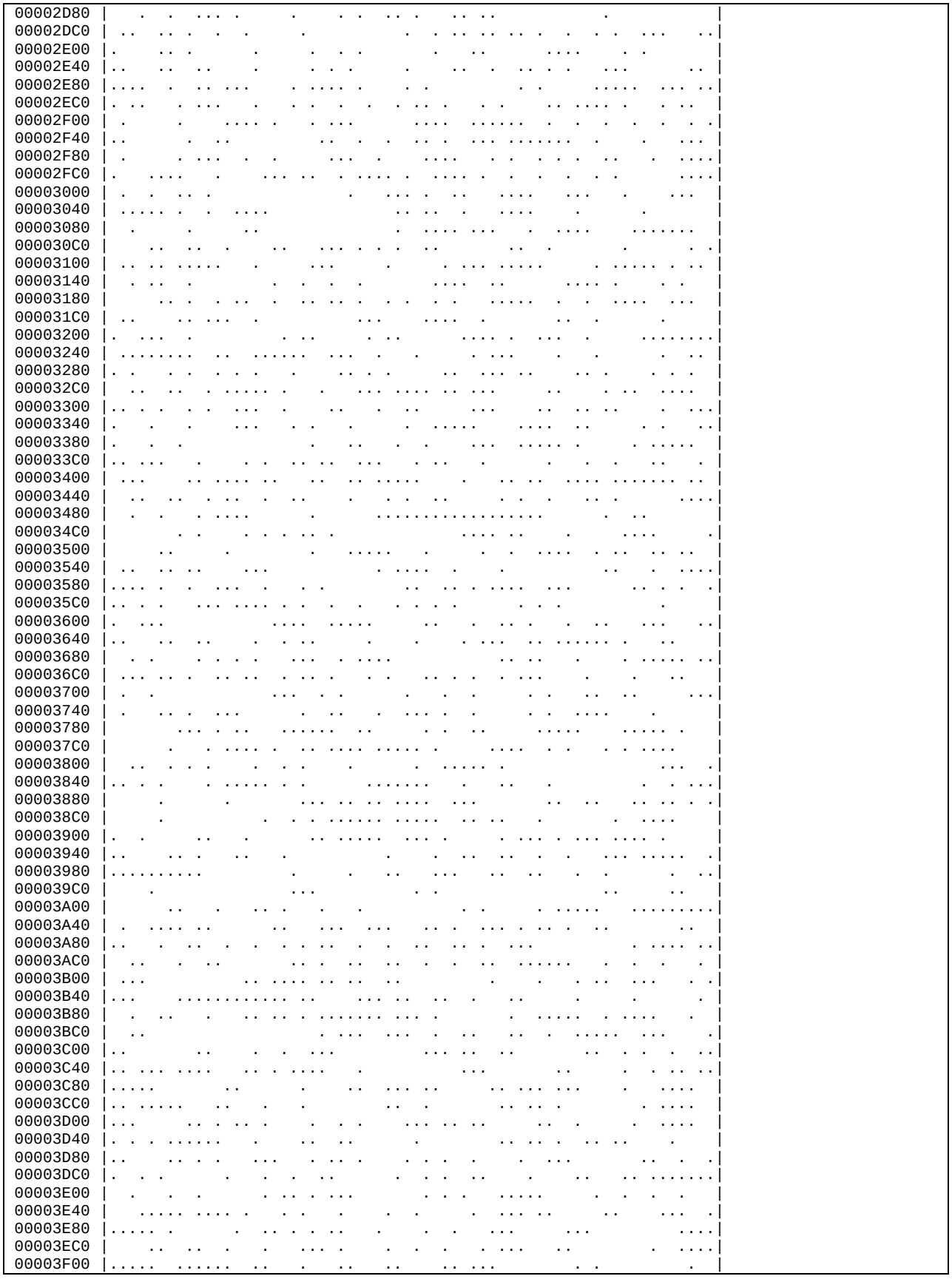
So in simple terms, the following legend will help as you peruse thru the comparison table:

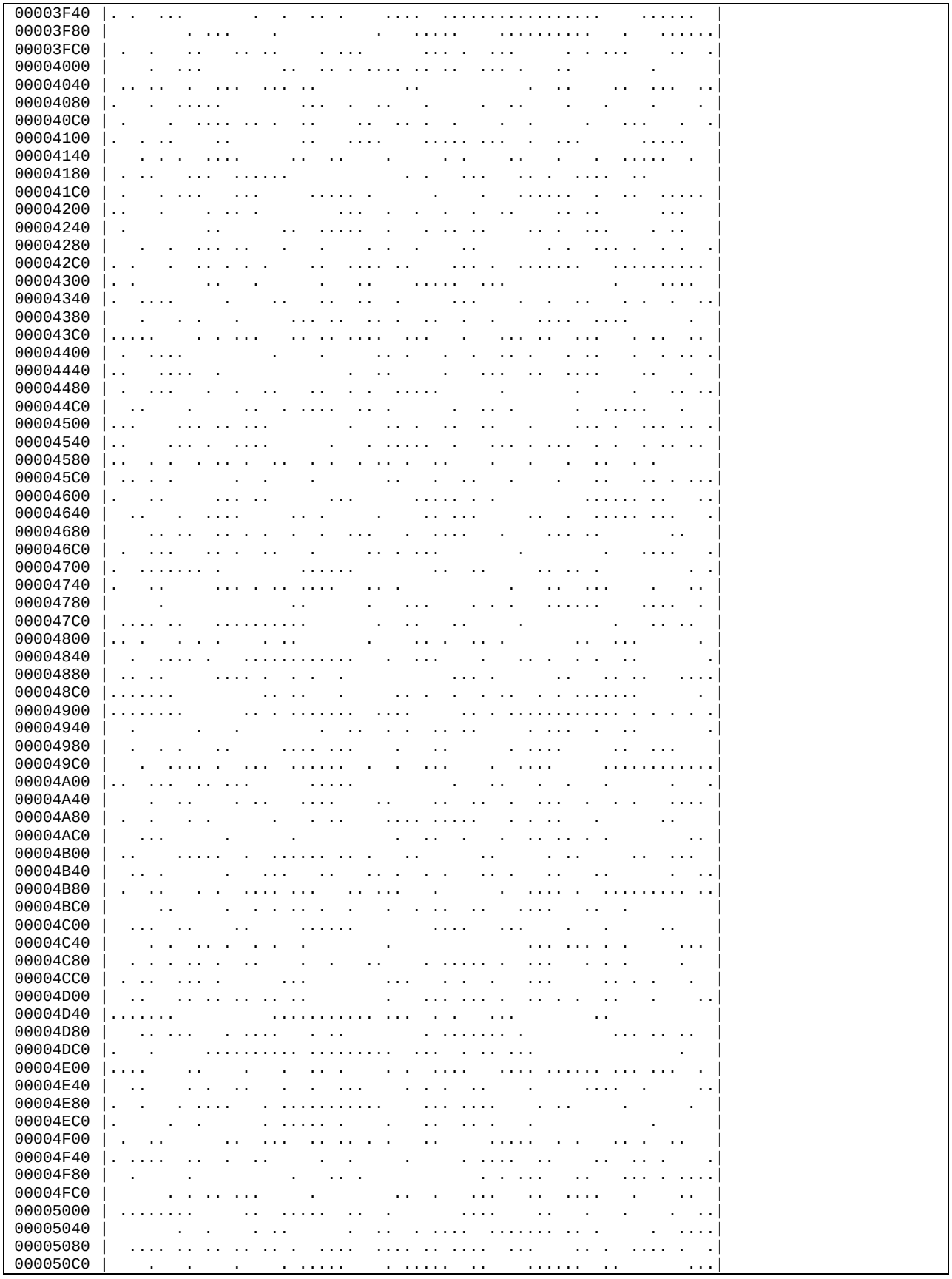
- A large section of **dots and spaces** indicate no matches;
- A section of **stars and dots** indicate a successful match;
- The initial match (memory 00000000 – 00001000) indicates a Windows executable and is common for many programs;
- The final match (beginning around 0001A1C0) indicates shared libraries;
- The significant matches in the middle (such as 0000A780 – 00014680 and 000164C0 – 00016C00) suggest common source code.

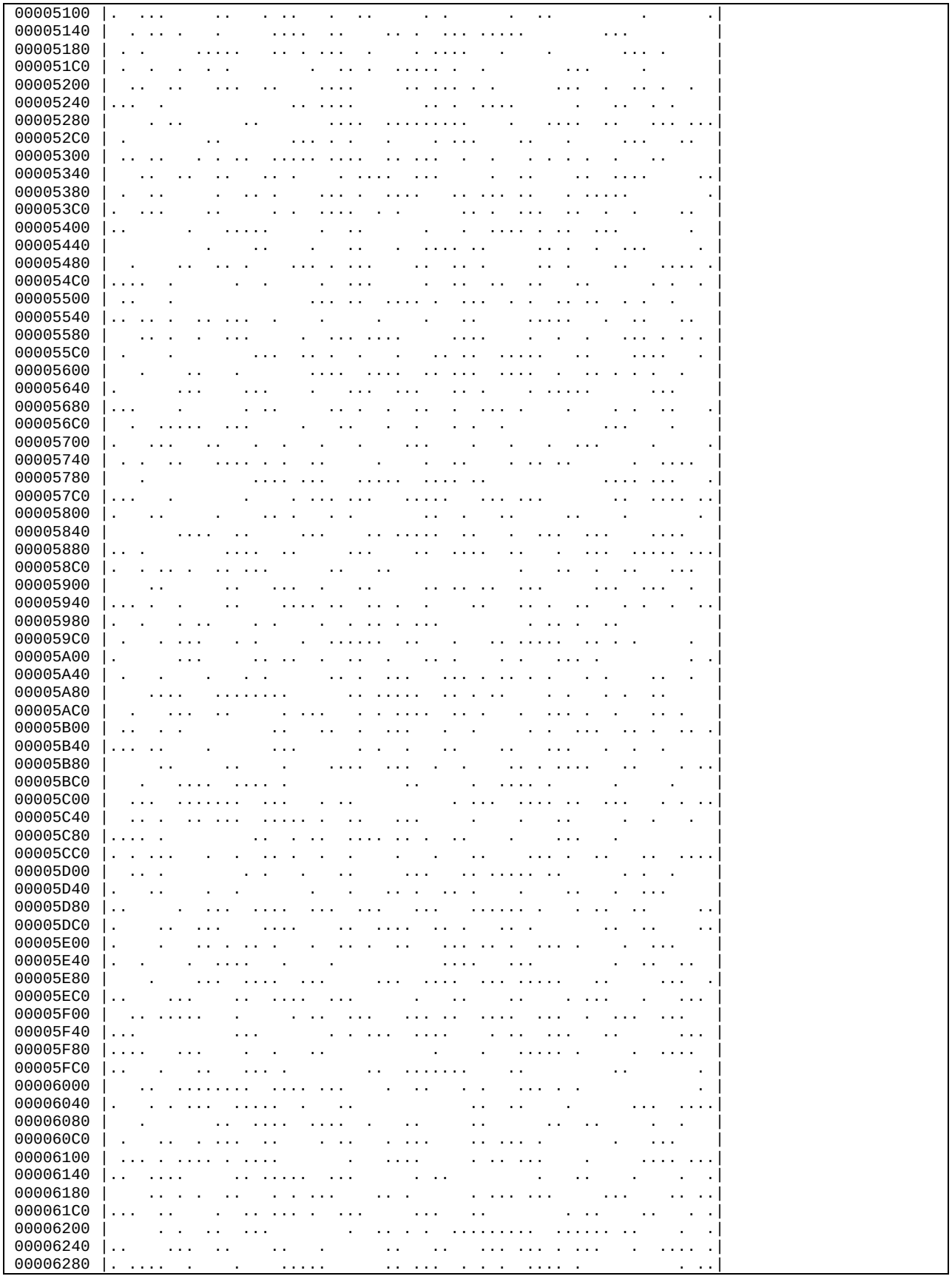


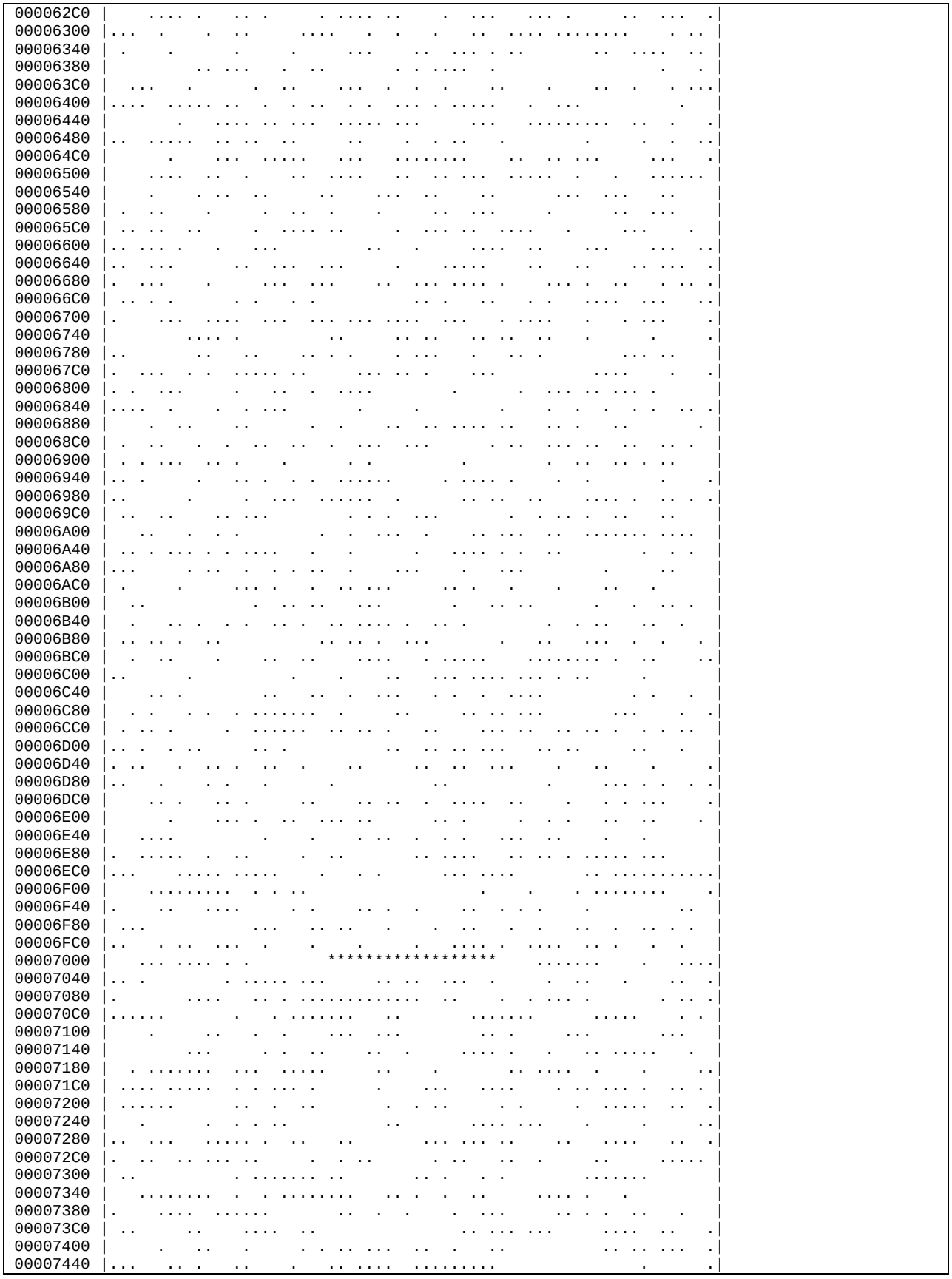


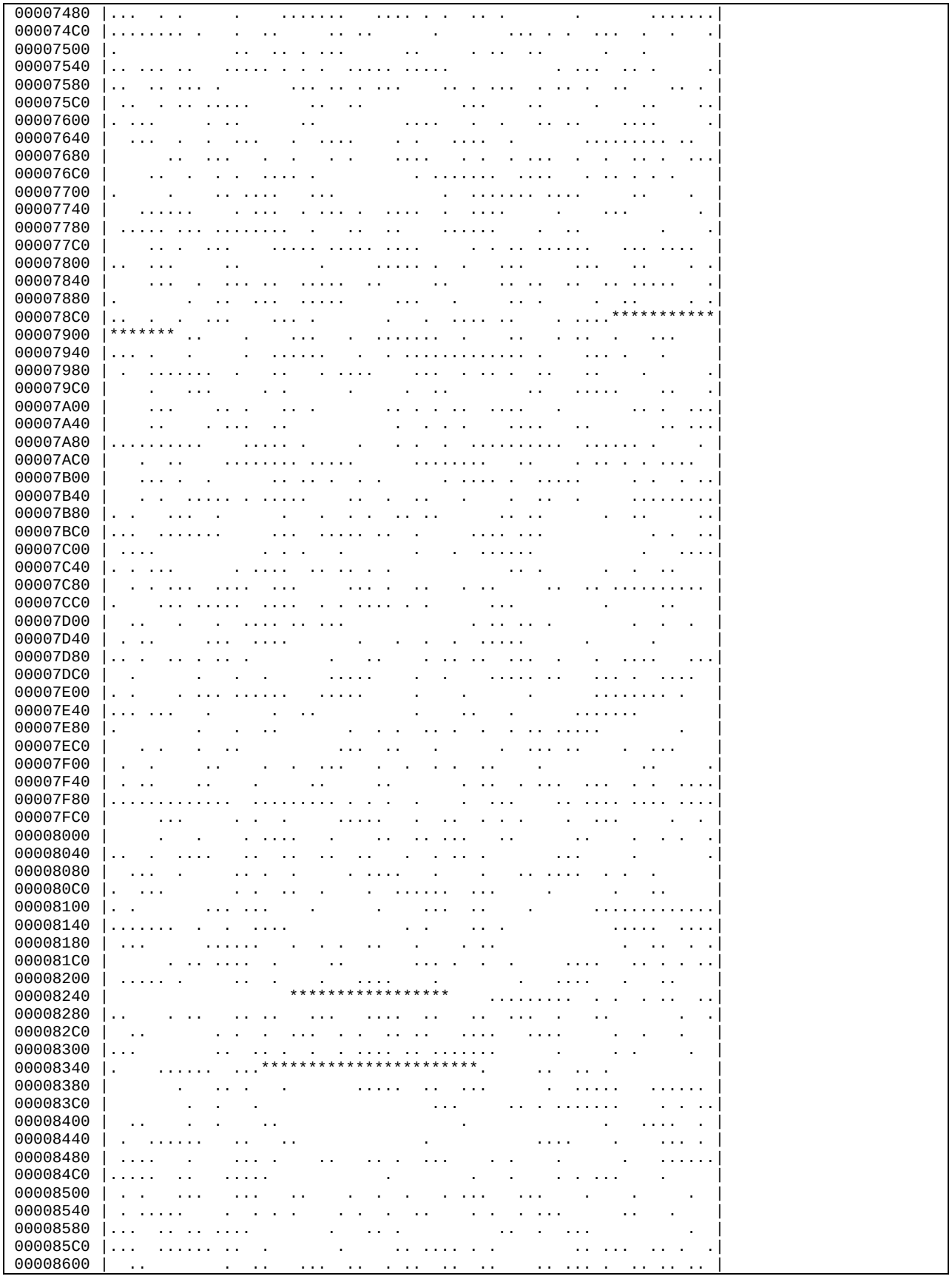


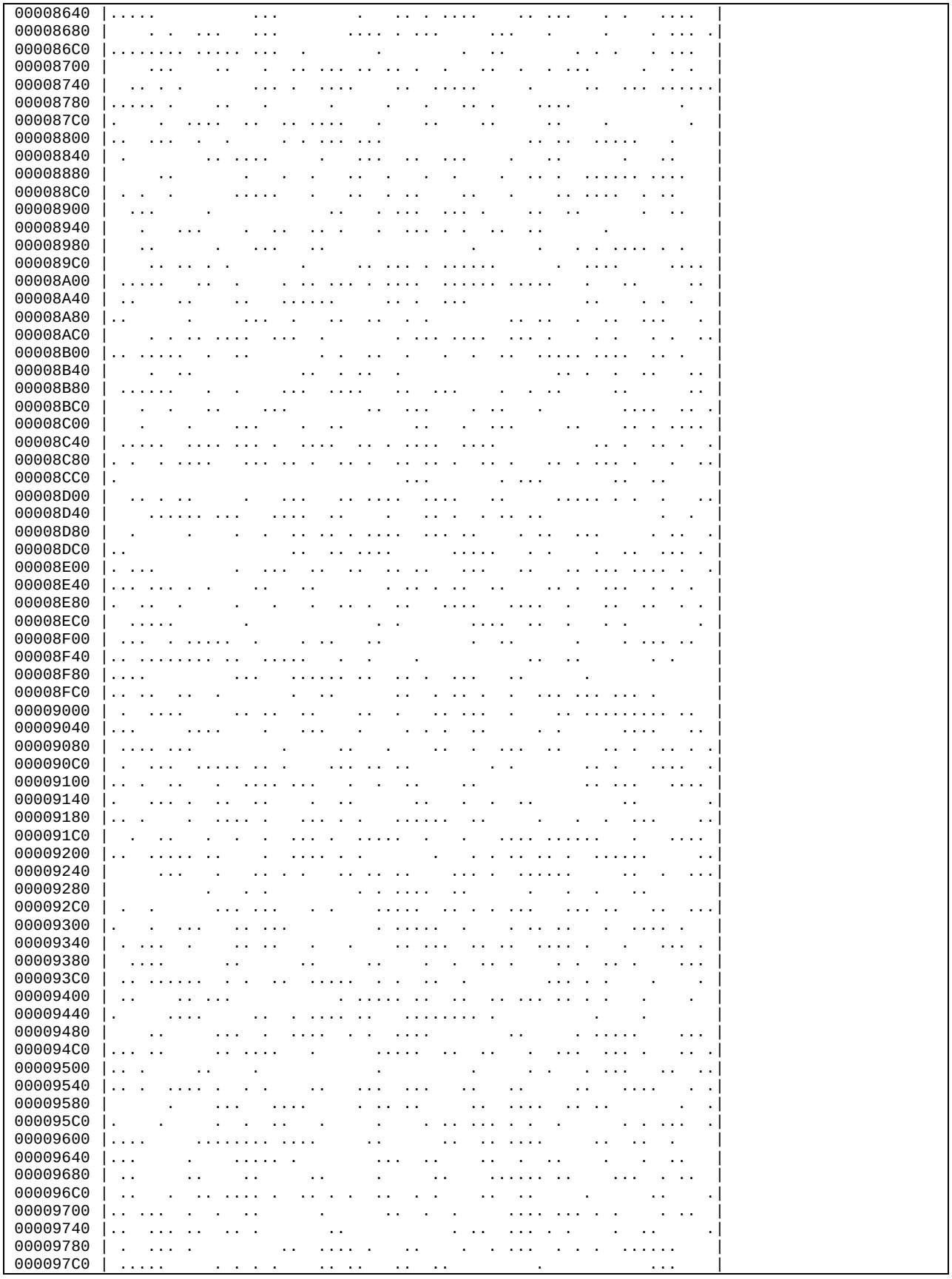


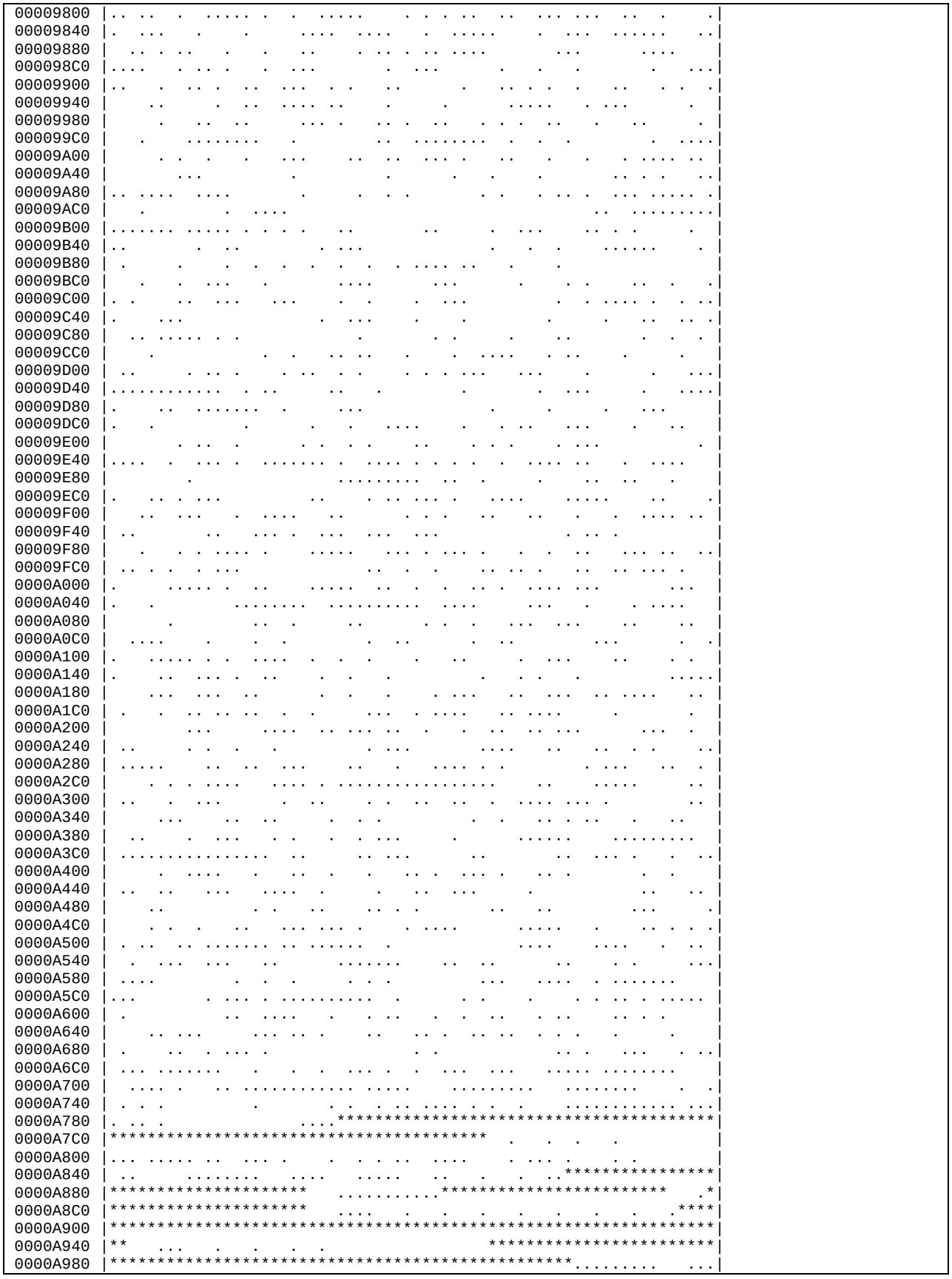


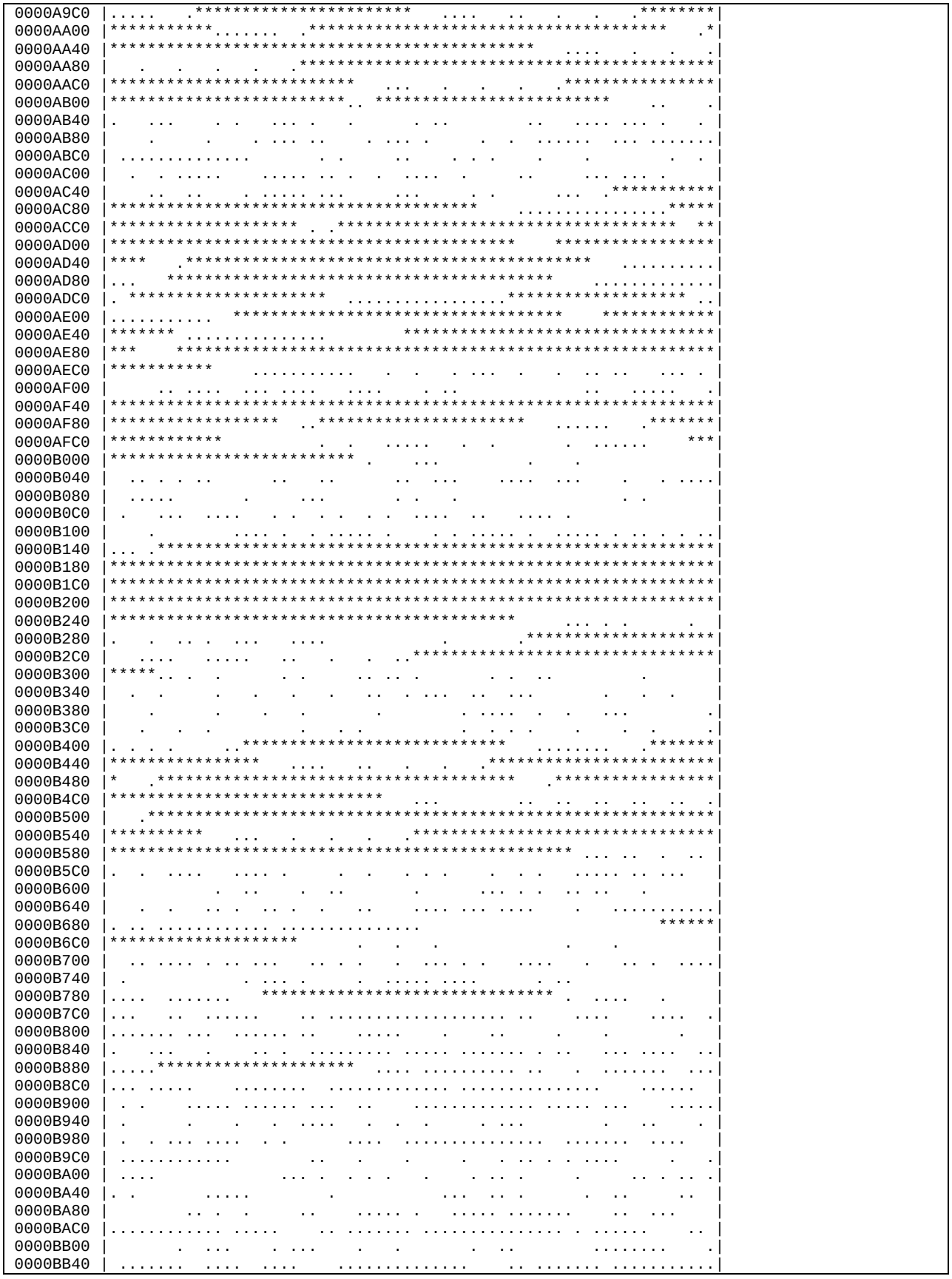


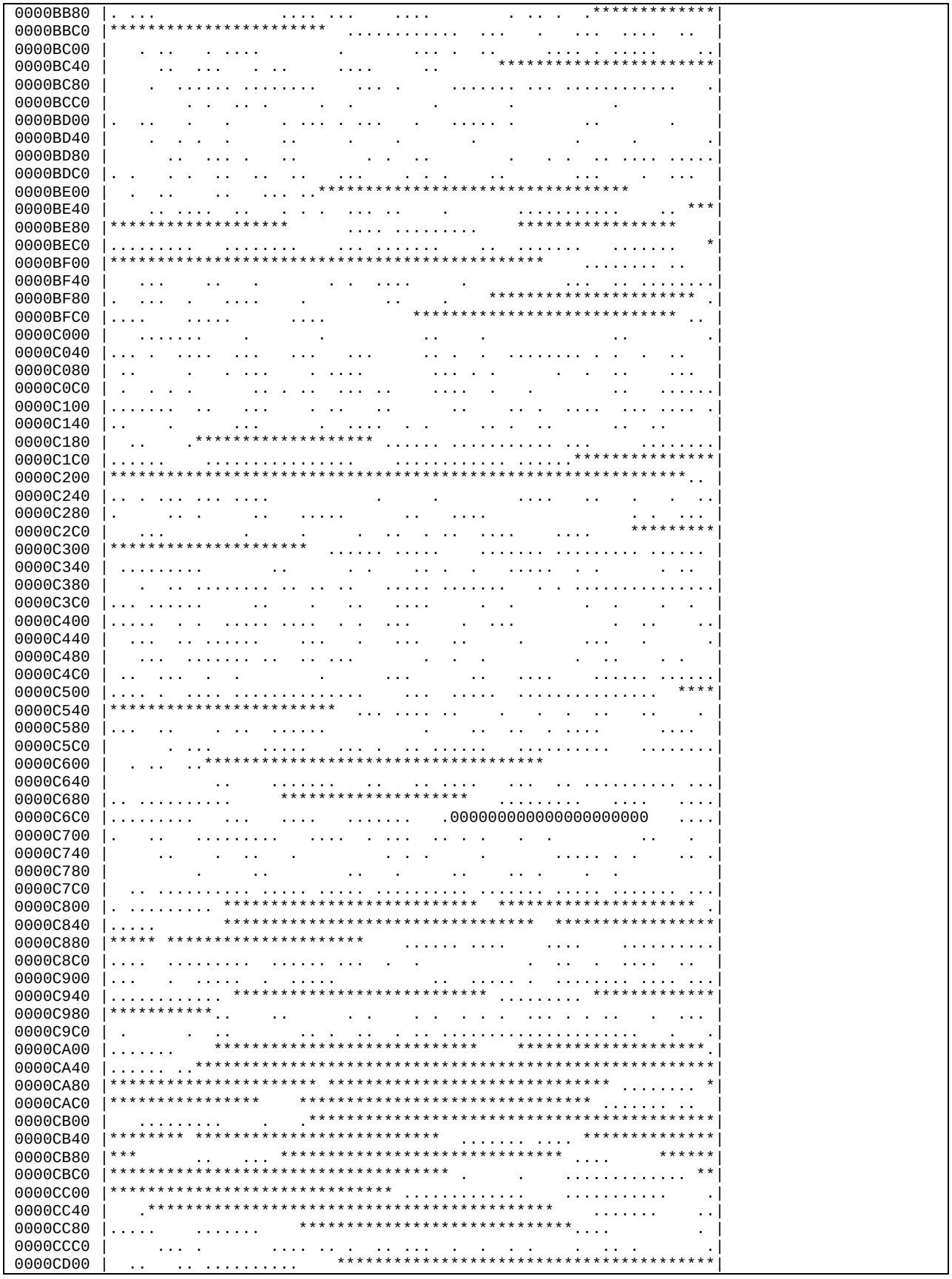


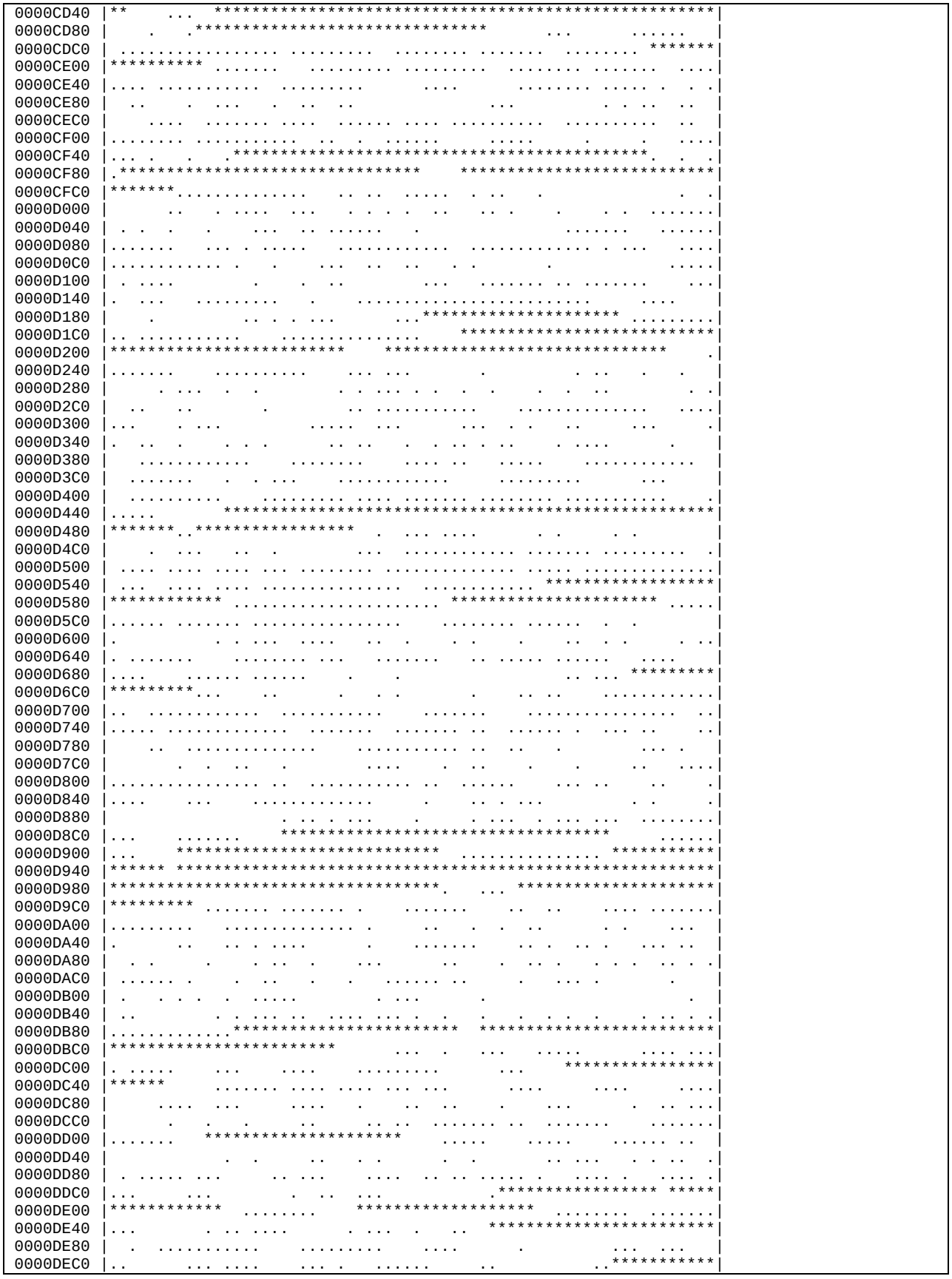


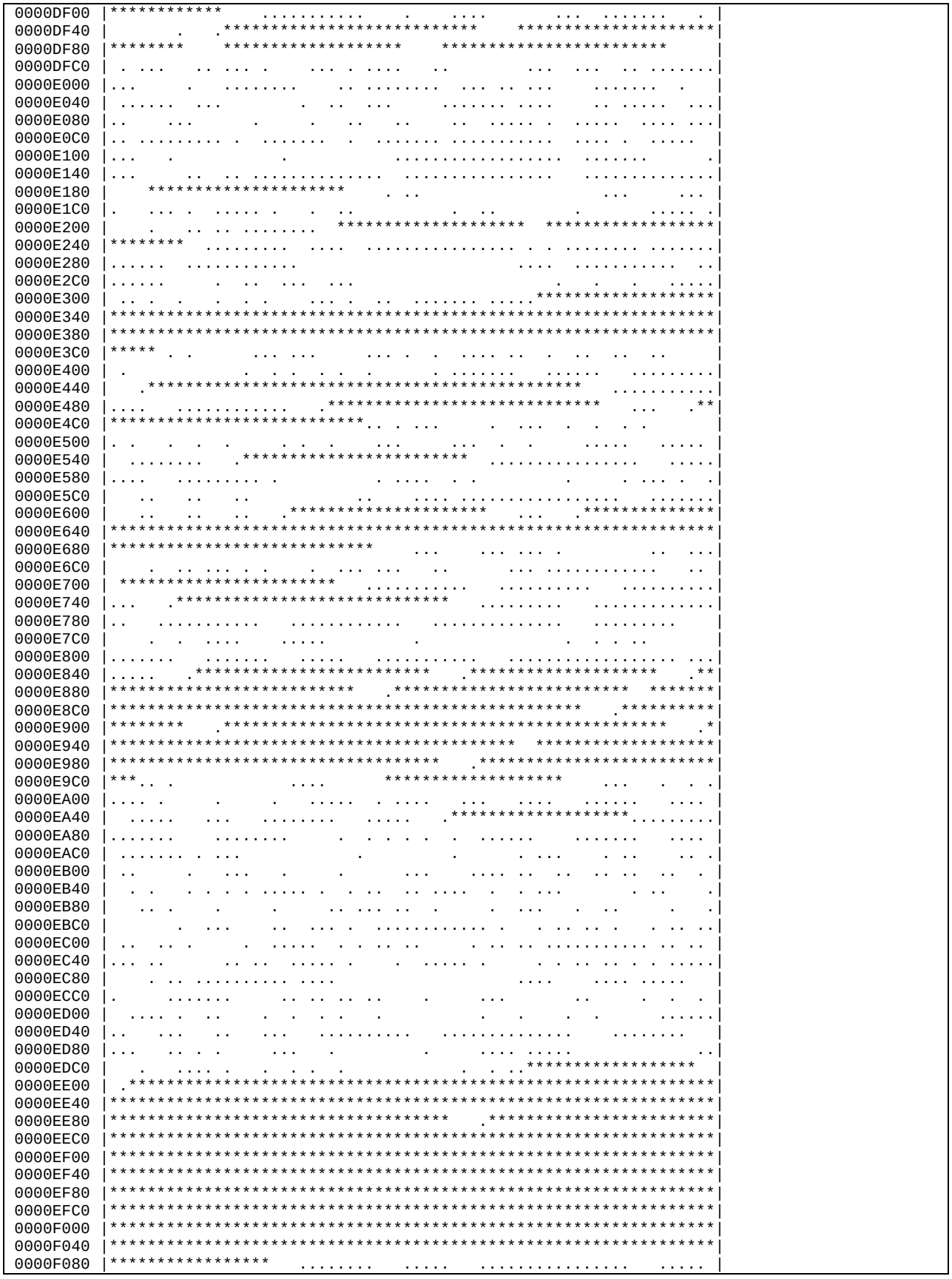


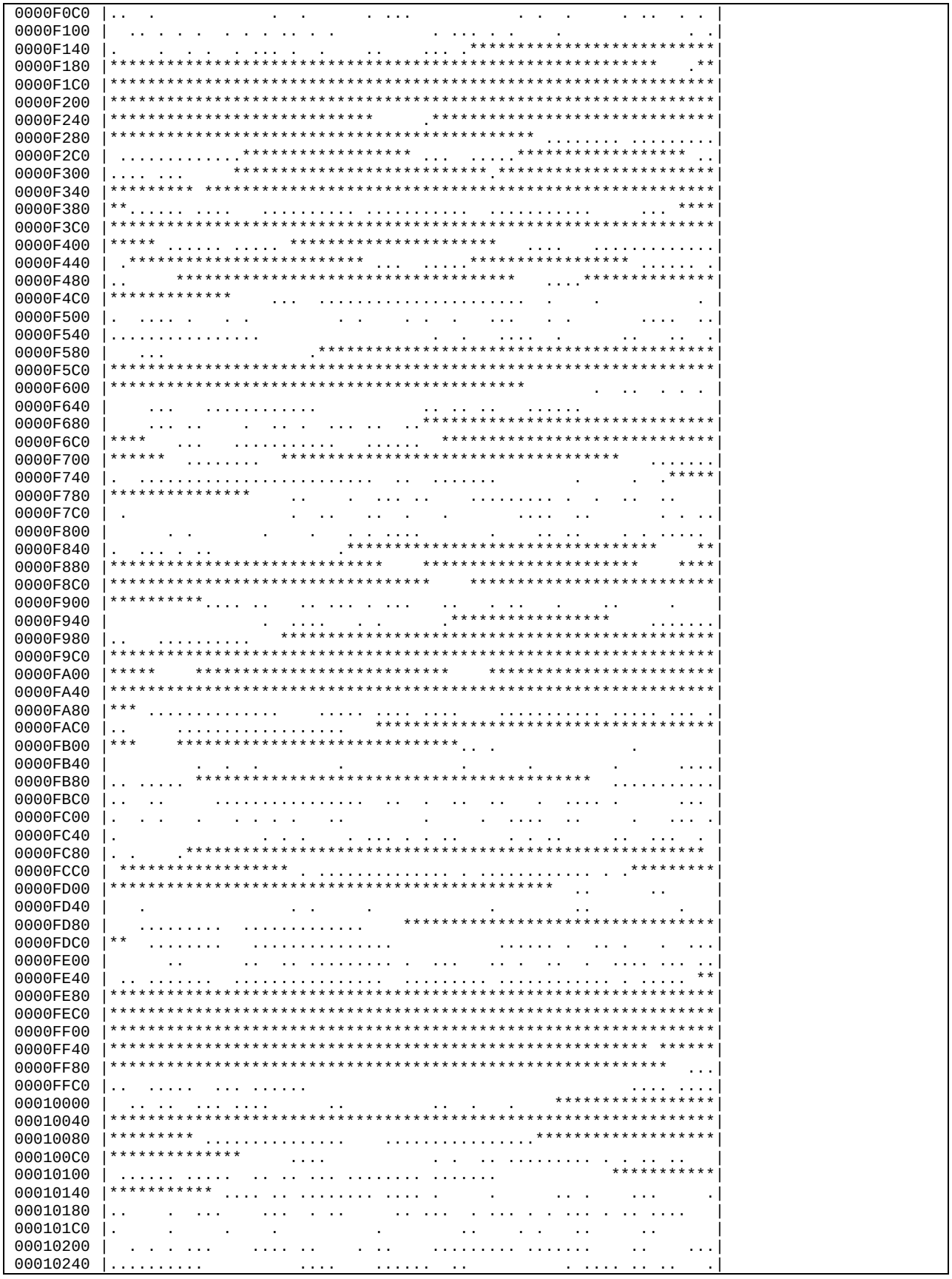


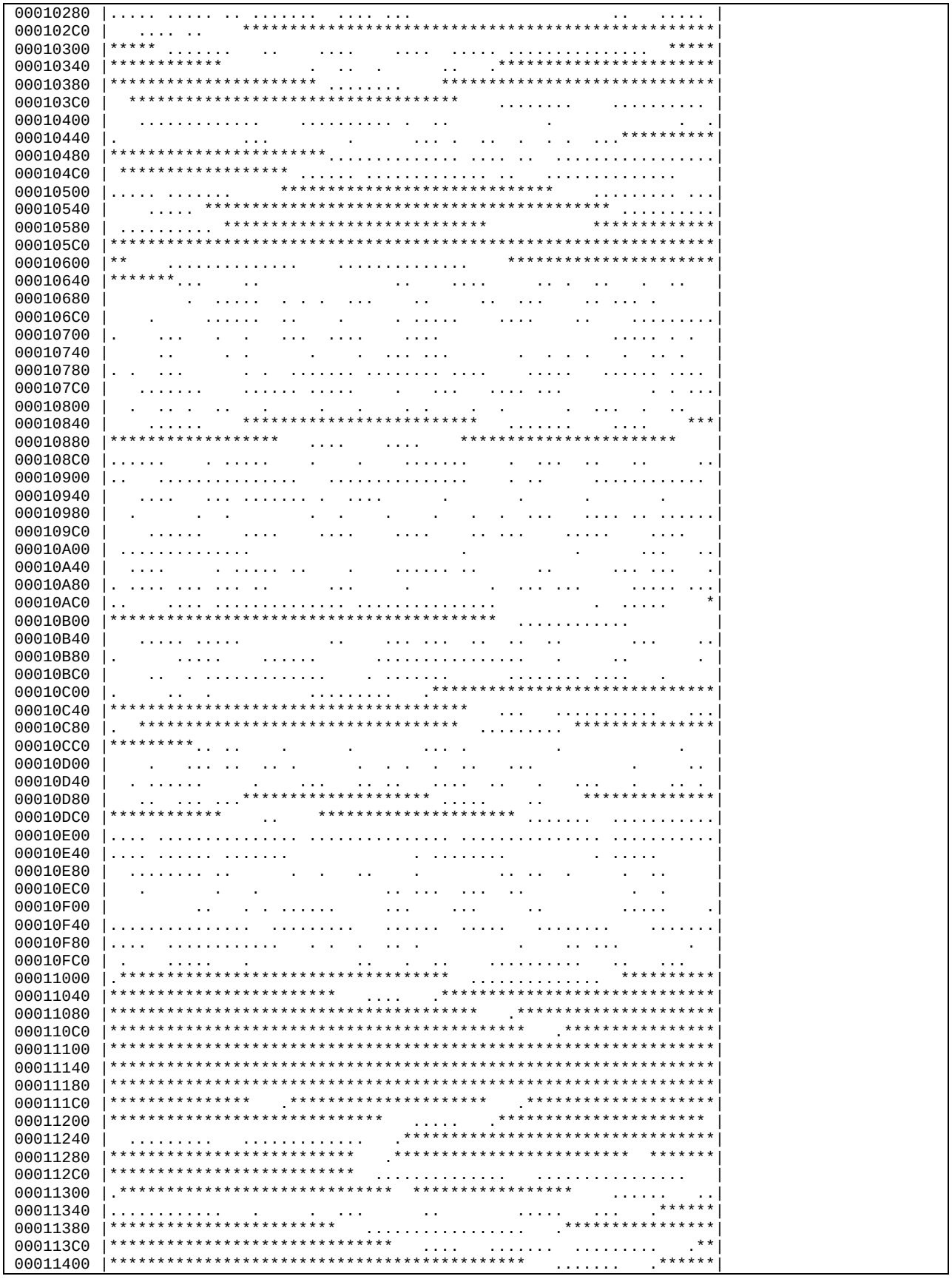


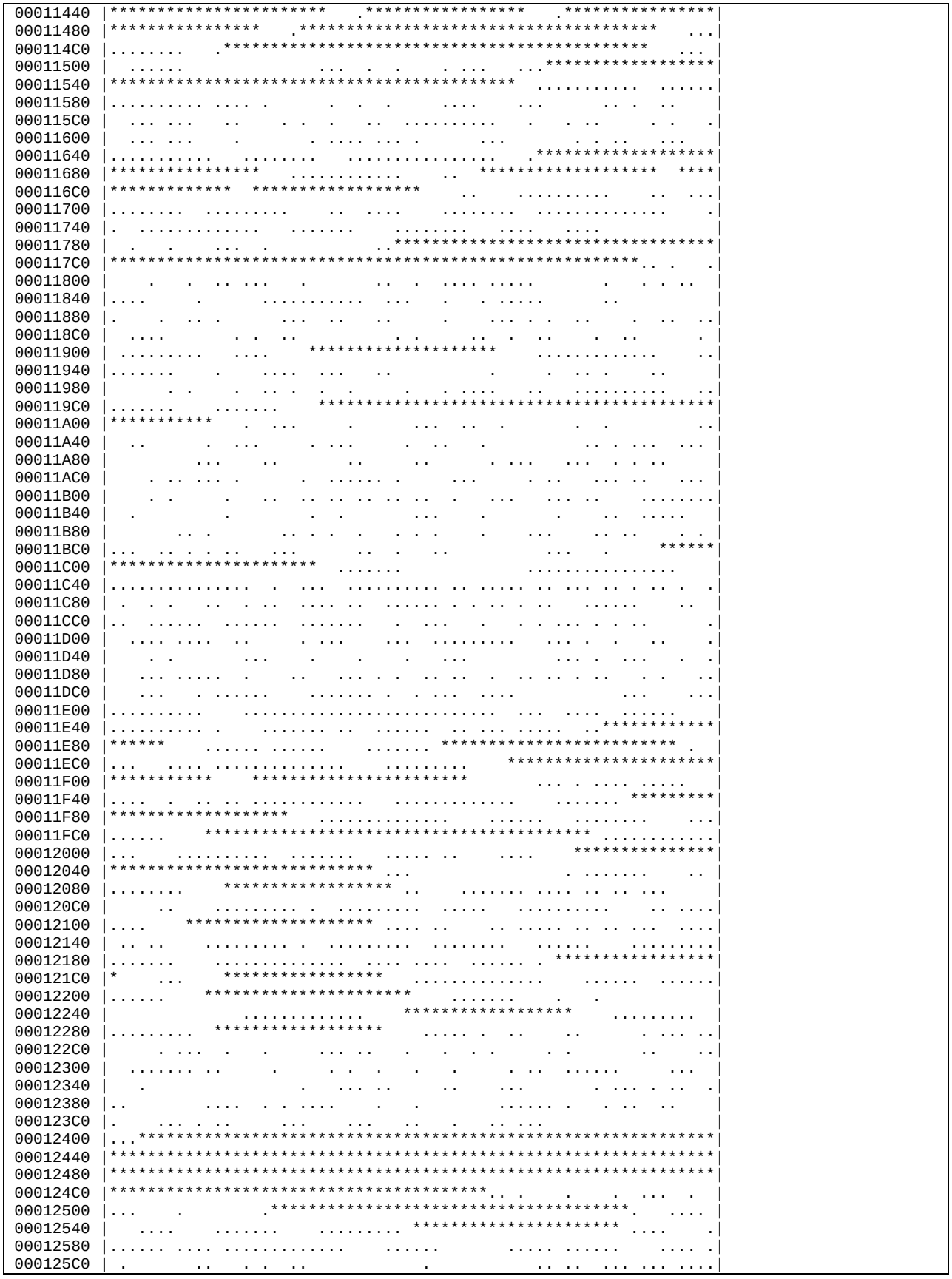


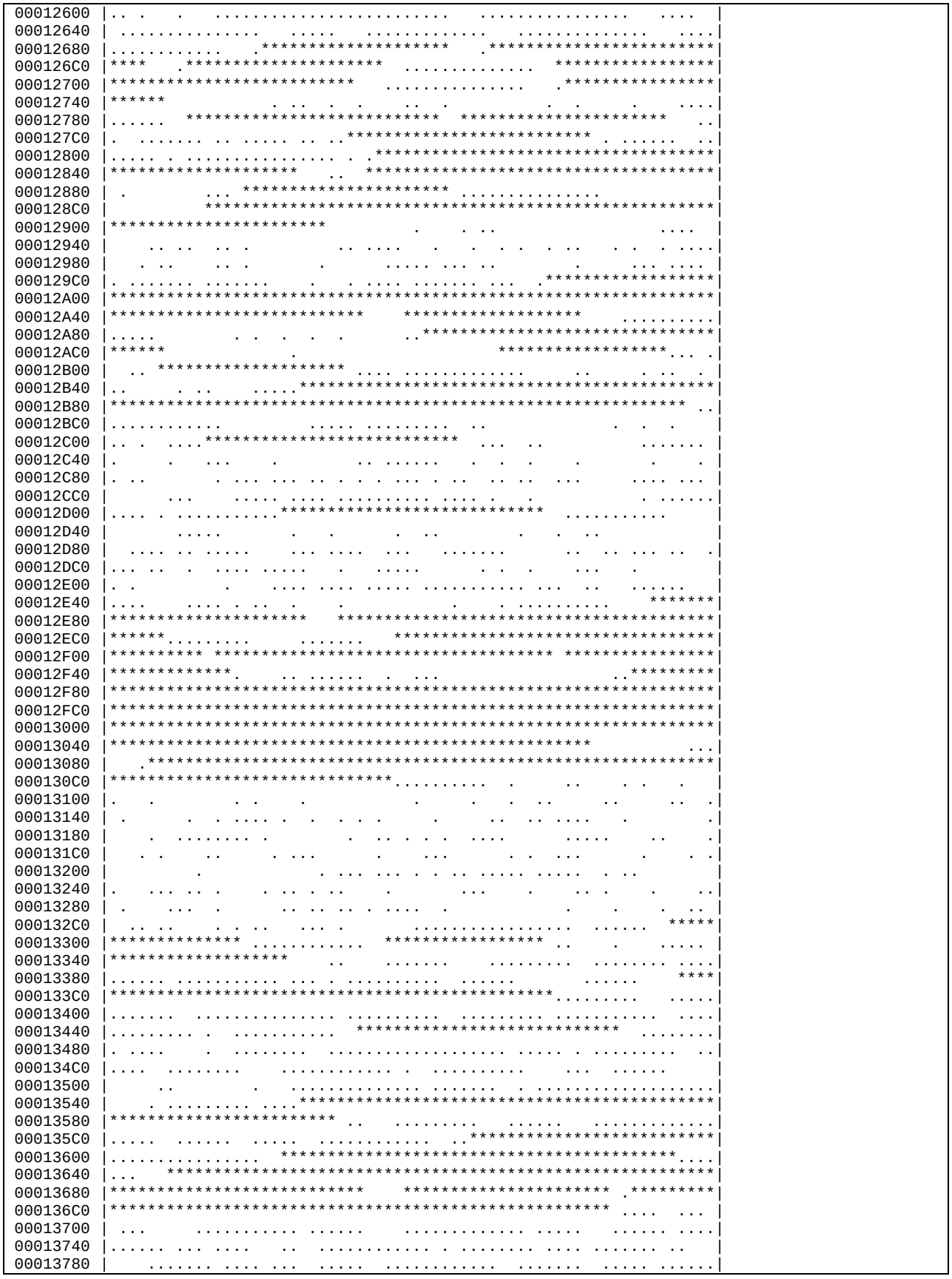


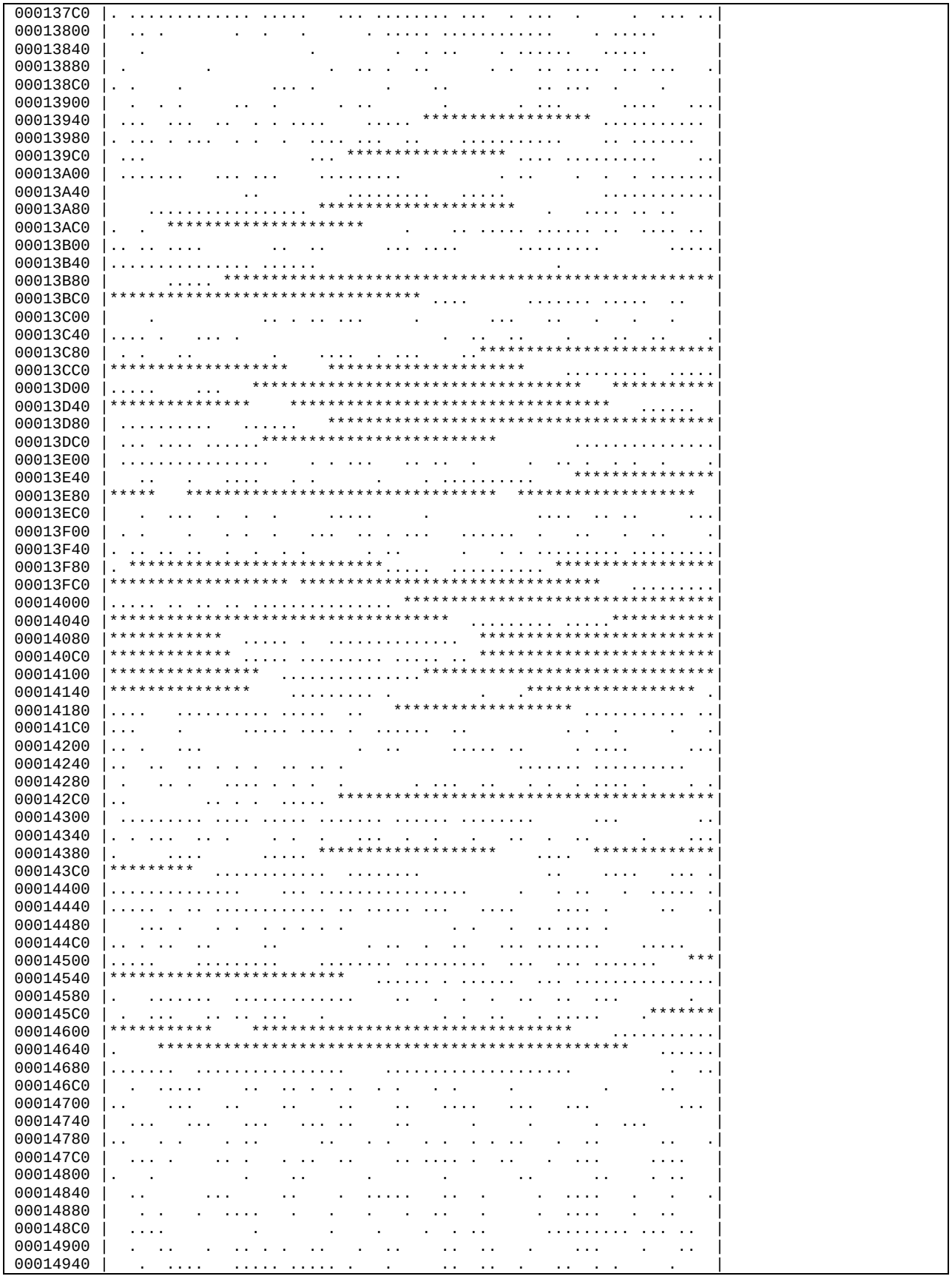


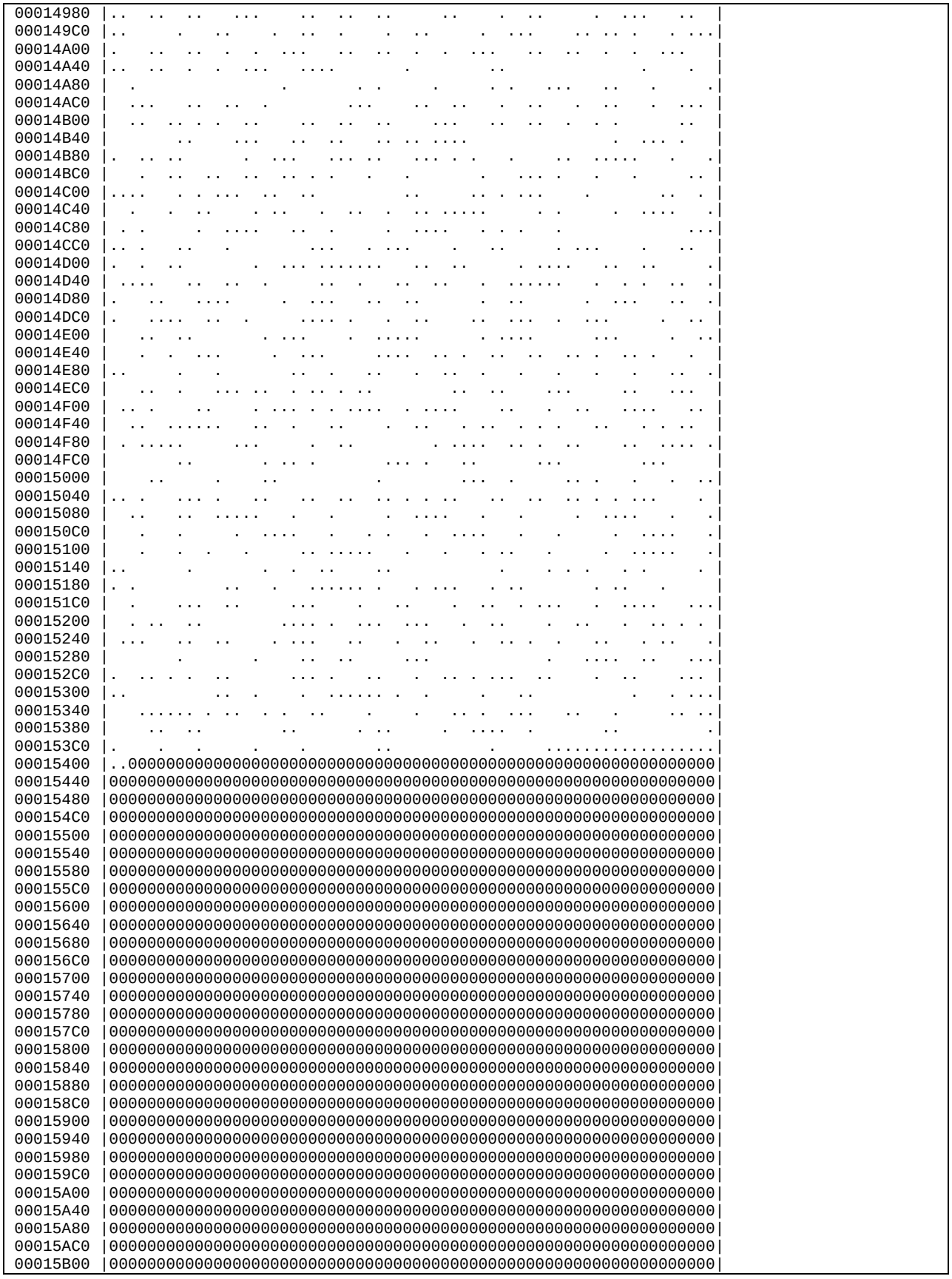


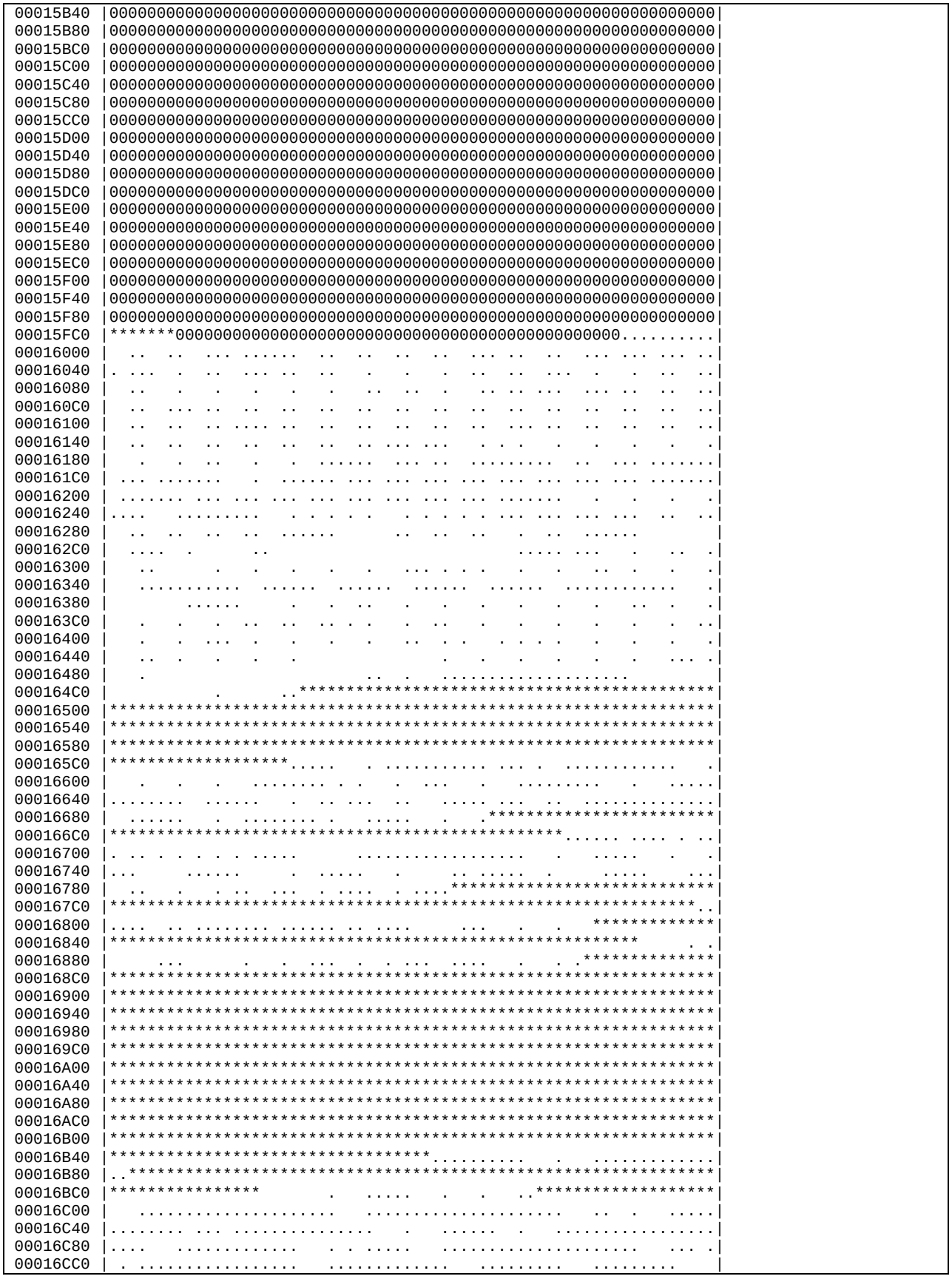


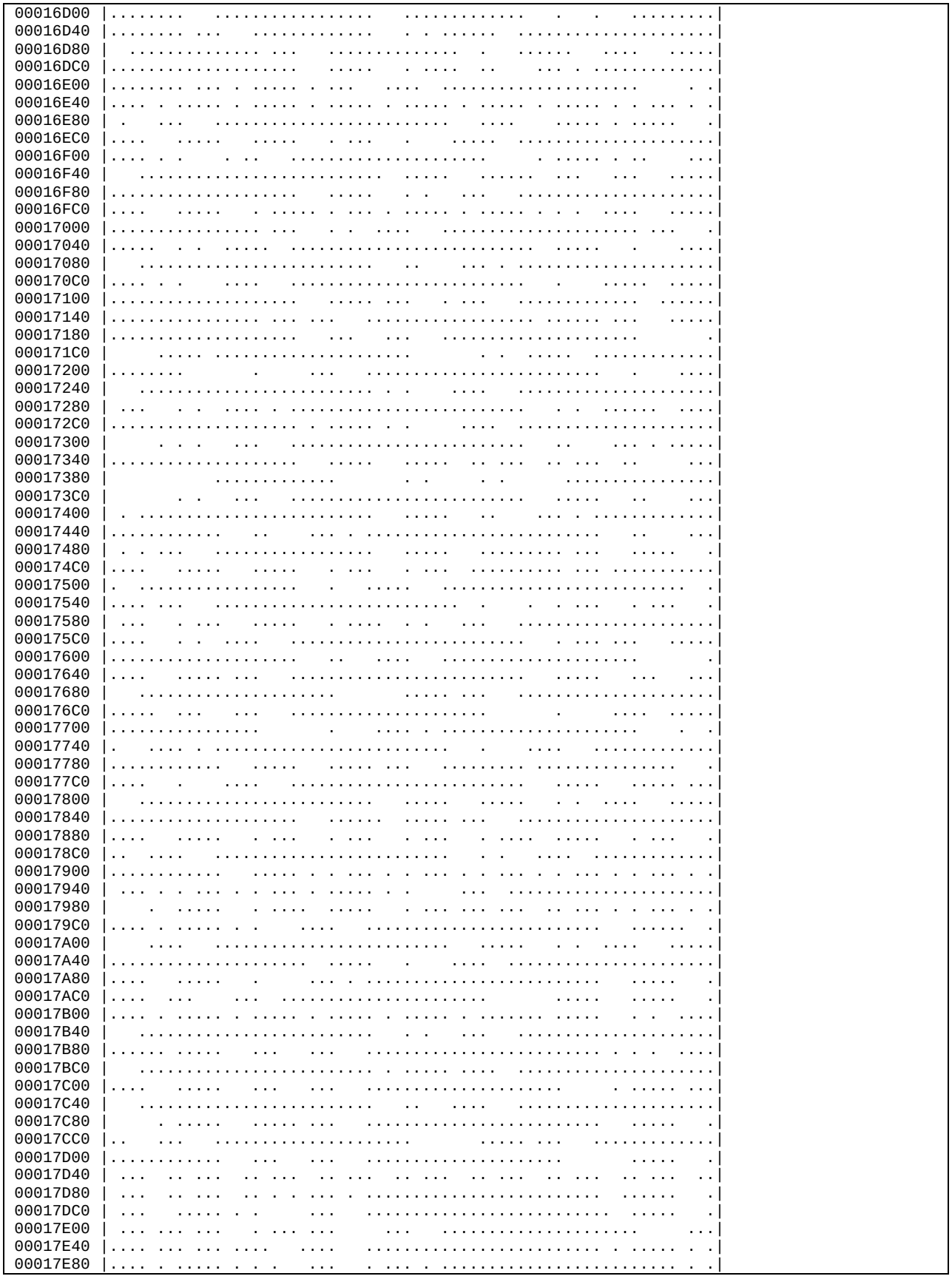


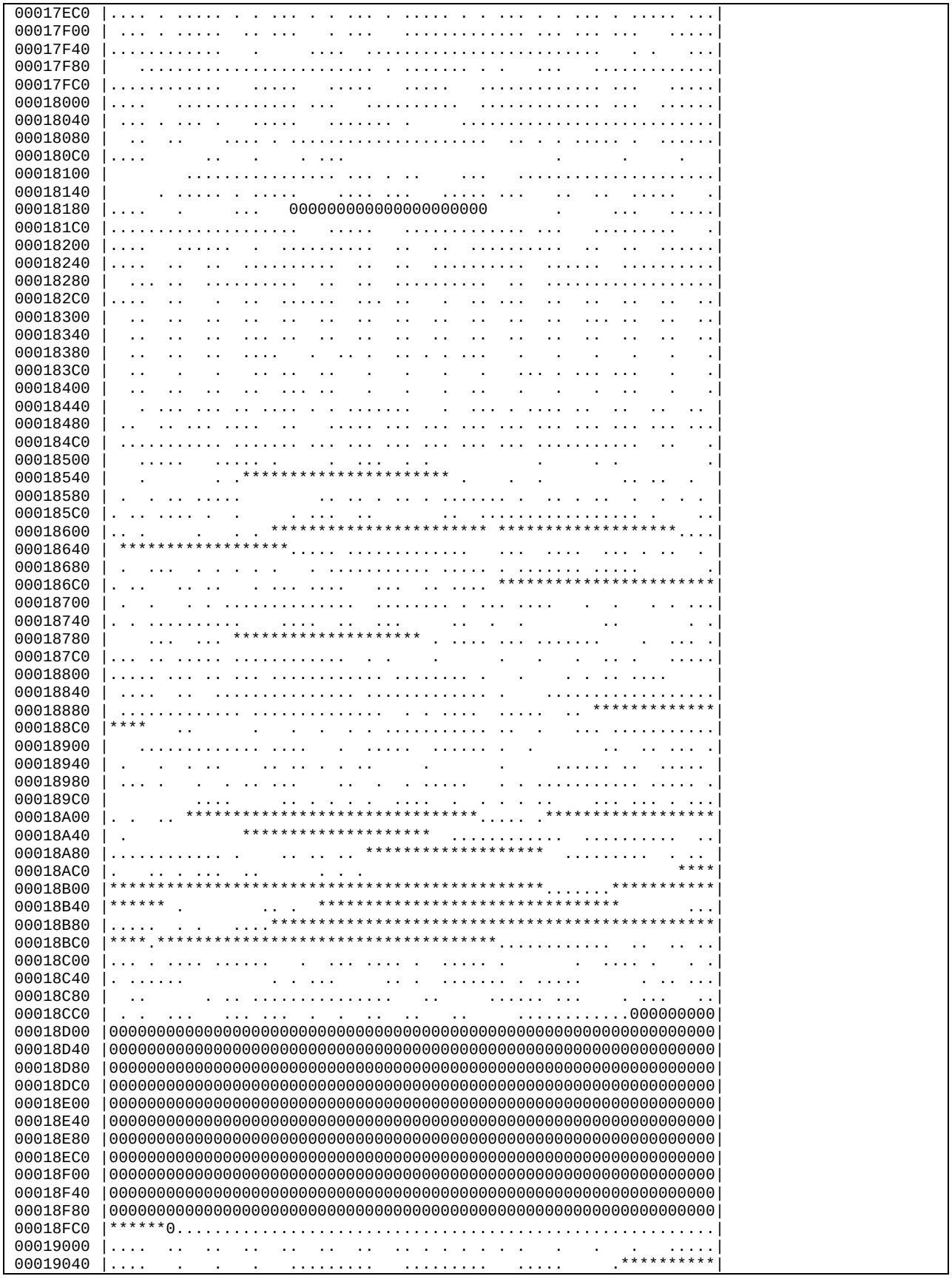











00017EC0
00017F00
00017F40
00017F80
00017FC0
00018000
00018040
00018080
000180C0
00018100
00018140
00018180
000181C0
00018200
00018240
00018280
000182C0
00018300
00018340
00018380
000183C0
00018400
00018440
00018480
000184C0
00018500
00018540
00018580
000185C0
00018600
00018640
00018680
000186C0
00018700
00018740
00018780
000187C0
00018800
00018840
00018880
000188C0
00018900
00018940
00018980
000189C0
00018A00
00018A40
00018A80
00018AC0
00018B00
00018B40
00018B80
00018BC0
00018C00
00018C40
00018C80
00018CC0
00018D00
00018D40
00018D80
00018DC0
00018E00
00018E40
00018E80
00018EC0
00018F00
00018F40
00018F80
00018FC0
00019000
00019040