A Comprehensive Measurement Study of Domain Generating Malware 原文翻译

**摘要：**

近年来，现代僵尸网络广泛采用了域生成算法（DGA）。主要目标是生成大量的域名，然后使用一小部分用于实际的C&C通信。这使得DGA对于botmasters来说非常有吸引力，可以强化僵尸网络的基础设施，并使其能够抵御黑名单和攻击，比如攻击。虽然早期的DGA被用作备份通信机制，但一些新的僵尸网络将其作为主要的通信方式，因此对DGA的详细研究显得尤为重要。

在本文中，我们通过分析43个基于DGA的恶意软件家族和变体，对DGA环境进行了全面的度量研究。我们还提出了DGAs的分类法，并用它来描述和比较所研究的家族的性质。通过重新实现这些算法，我们预先计算了它们生成的所有可能的域，覆盖了大多数已知的和活动的DGA。然后，我们研究了超过1800万个DGA域的注册状态，表明通过预先计算未来的DGA域，可以可靠地识别出相应的恶意软件家族和相关活动。我们还将深入了解僵尸主机在域注册方面的策略，并找出以前基于DGA的僵尸网络的一些缺陷。我们将为将来的研究共享数据集，还将提供一个web服务来检查域中潜在的DGA身份。

**简介：**

僵尸网络是由受恶意软件影响的机器（bot）组成的网络，由对手（botmaster）通过命令和控制（C&C）通信通道进行远程控制。僵尸网络已成为网络犯罪分子进行恶意活动的主要手段，如发起拒绝服务攻击、发送垃圾邮件和窃取个人数据。最近的研究表明，一些僵尸网络由100多万个机器人组成[40]，说明了它们的威胁程度。

执法和安全研究人员经常试图通过执行takedown尝试来破坏活跃的僵尸网络。这些攻击的主要目标是僵尸网络的C&C通信基础设施。这些攻击的一个突出例子是sinkholing，所有的机器人都被重定向到一个被攻击者控制的机器上，这个机器被称为sinkhole。因此，机器人程序将无法与原始C&C服务器通信。为了应对这些措施，botmaster开始发明新技术来保护僵尸网络的基础设施。近年来得到广泛应用的一个重要方法是使用域名生成算法。

域名生成算法（DGA）用于动态生成大量貌似随机的域名，然后选择其中的一小部分进行C&C通信。生成的域名是基于给定的种子计算的，种子可以由数值常量、当前日期/时间甚至Twitter趋势组成。种子作为botmasters和机器人之间的共享秘密来计算共享的集合点。通过不断地改变使用的域名，依赖静态域黑名单的检测方法变得无效。此外，通过动态生成域名，botmasters不必在其恶意软件二进制文件中包含硬编码的域名，这使得提取这些信息变得复杂。同时，由于不同的域名在不同的时间点被观察到，使得生成的域依赖于时间降低了从动态恶意软件分析系统中提取的域的价值。使用短期域名的另一个好处是在域名生效前不久就注册了，这是为了避免域名信誉服务。

DGAs的使用在攻击者（botmasters）和防御者（安全研究人员和执法人员）之间造成了高度不对称的局面。Botmasters需要访问一个域名来控制或迁移他们的僵尸主机，而防御者需要控制所有的域名以确保成功地takedown。由于有1000多个顶级域（TLD）[9]可供选择，攻击者很容易在全球范围内分散对域的责任，迫使防御者进行额外的协调与合作。例如，臭名昭著的Conficker僵尸网络（C版）的DGA每天生成50000个域名，这些域名分布在113个TLD上。这需要包括ICANN在内的30个不同组织的全球合作努力来遏制这一威胁。

在这项工作中，我们通过分析43个基于DGA的恶意软件家族和变体，对DGA环境进行了全面的测量研究。我们的分析是基于这些家族的DGA逆向工程。我们提出了一个分类法来描述DGAs的主要方面，并用它来描述和比较所研究的dga。此外，我们重新实现了所有这些DGA，并基于之前和正在进行的恶意活动中使用的253个种子集计算了所有可能的输出。然后我们用这组数据来识别在过去14年里收集的90亿条WHOIS记录中由DGA生成的域。我们分析了这些域的注册状态及其所有权的变化，这些变化通常表示从恶意域过渡到sinkhole域。这使我们能够描述botmasters和sinkhole操作符的注册行为，并详细研究DGA域的生存期

据我们所知，我们的工作反映了现代僵尸网络所采用的第一次对DGA全景式的系统研究。安全研究人员之前已经研究过DGA，并提出了检测和集群基于DGA的恶意软件的方法[13,18,43,54]。相反，我们的研究是以一种自下而上的方式对DGA进行深入分析的基础上，通过重新实现算法并列举它们生成的全套域名，使我们能够对DGA生成域名有一个真实的标注数据，而不会出现误报。

总之，我们作出以下贡献：

•我们提出了DGA的分类法来描述和比较其特性。

•我们分析了43个恶意软件系列和变种的DGA。利用253个已识别的种子，我们列举了由这些算法产生的所有可能的域名，覆盖了大多数已知的和活跃的DGA。

•我们研究了涵盖8年的1800万个DGA生成域名的注册状态，并表明通过预先计算未来的DGA域名，可以可靠地识别相应的恶意软件家族和相关活动。

•我们分析了botmasters和sinkholers在域注册方面的策略，并确定了以前基于DGA的僵尸网络的几个缺陷。

•我们共享数据集以备将来研究，并提供一个名为DGArchive[38]的web服务来检查查询的域是否来自DGA。

2 A DGA分类法

在本节中，我们将为DGA提出一个分类法来描述其属性并进行比较。为了对不同属性的整个频谱进行建模，我们提出了两种特征来捕获域生成算法的不同方面。然后将这两个特性组合在一个将DGA分类为类的分类法中。

2.1Seed Source

种子作为计算生成域所需的共享机密，也称为算法生成域（AGD）[54]。它是执行域生成算法所需的参数集合。典型参数包括数值常量（例如，伪随机数生成器的域或种子的长度）或字符串（例如，字母表或可能的TLD集）。

Seeding的两个特性对于描述DGA具有重要意义（cp.Barabosch等人[16]）：

**时间相关性**是指DGA包含一个时间源（例如，受入侵主机的系统时间或HTTP响应中的日期字段）以计算AGD。因此，生成的域将只有一个有效期，在此期间，这些域将被受入侵的系统请求。

**确定性**强调参数的可观测性和可用性。对于大多数已知的DGA，DGA执行所需的所有参数都是已知的，可以计算出所有可能的域。两个DGA使用时间不确定性，通过使用不可预测但可公开访问的数据进行播种，从而禁止对未来AGD的任意预测。恶意软件家族Bedep[44]利用欧洲央行（ecb）每日公布的外汇参考汇率，而Torpig[50]的一个后来的变种则使用Twitter trends进行播种。在这两种情况下，这只会导致一旦用于seeding的不可预测的数据可用攻击者和防御者在每个活动时间窗口中竞争域的注册。但是，这并不妨碍历史分析，因为种子数据和由此生成的域仍然可以随着时间的推移而收集。赛门铁克发现了另一种不确定性[45]。他们对Jiripbot的分析显示，恶意软件会过滤一组系统属性，包括MAC地址和硬盘驱动器卷ID，使攻击者可以在DGA种子中使用这些属性。在这种情况下，系统信息被认为是不确定的，攻击者在入侵之前，也从来没有公开可见。

时间依赖性和确定性允许以下四种组合：时间无关与确定性（TID）、时间相关与确定性（TDD）、时间相关与非确定性（TDN）、时间无关与非确定性（TIN）。在我们的数据集中，我们只观察到使用前三类种子属性的DGA。

2.2域名生成方案

在分析过程中，除了种子的特性外，还出现了4种不同的生成方案。

**Arithmetic-based DGAs**基于算术的DGA计算一系列值，这些值要么具有可用于域名的直接ASCII表示，要么在一个或多个硬编码数组中指定偏移量，构成DGA的字母表。它们是最常见的DGA类型。

**Hash-based DGAs**基于Hash的dga使用Hash的hexdigest表示来生成AGD。我们使用MD5和SHA256来识别dga来生成域名。

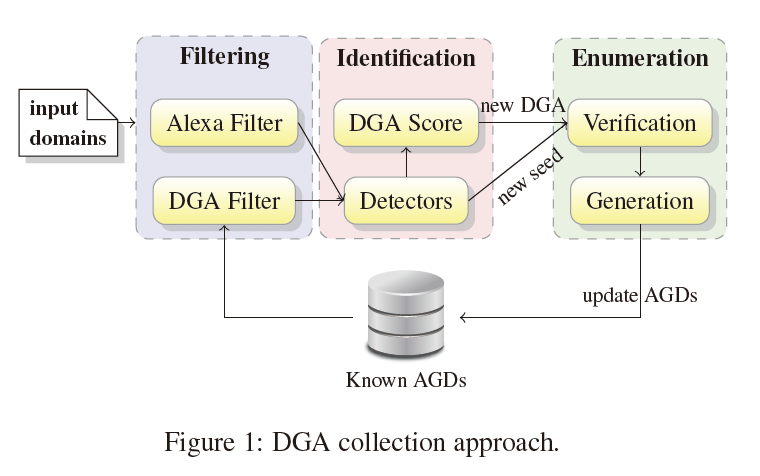
**Wordlist-based DGAs** 基于单词表的dga将连接一个或多个单词列表中的单词序列，从而减少随机吸引力，从而增加伪装域。这些单词表要么直接嵌入恶意软件二进制文件中，要么从公共访问源获取。

**Permutation-based DGAs：**基于置换的分布式遗传算法通过对初始域名的置换来导出所有可能的AGD。

我们用各自的起始字母A、H、W或P来缩写生成方案。作为上述生成方案的一部分，一些DGA利用伪随机数生成器pseudo-random number generators (PRNGs)来生成域。从实现自己的PRNGs到使用众所周知的技术，如线性同余发生器linear congruential generators（LCGs）[36]

2.3 DGA类型

作为DGA类型，我们考虑种子属性和生成方案的组合，以组合缩写命名，例如“TID-A”表示时间无关、确定的DGA，使用基于算术的域生成方案。在16种可能的组合中，我们只观察到43种DGA在我们的数据集中使用的6种类型：TDD-A（20）、TID-A（16）、TDD-W（3）、TDD-H（2）、TDN-A（1）、TID-P（1）。



3 DGA恶意软件数据集

在本节中，我们将描述如何识别和收集使用DGA的恶意软件样本。然后，我们描述了我们对算法进行逆向工程、重新实现和评估其实现的工作。

3.1识别基于DGA的恶意软件

我们研究的第一步是收集一组具有代表性的基于DGA的恶意软件家族。为此，我们开发了一个系统，通过分析给定恶意软件样本生成的一组域名来自动识别潜在的新DGA。这有助于我们将手动逆向工程所需的恶意软件样本集最小化，从而将我们的工作集中在可能实现新DGA的示例上。图1显示了系统的高级概述。首先，我们过滤掉已知的AGD和良性区域。其次，我们识别由先前已知的DGA生成但具有新种子的域，以及可能由新DGA生成的域。第三，如果一个新的DGA或一个新的种子被识别出来，我们手工对相应的样本进行逆向工程，提取种子和算法，并计算它生成的域名集。下面，我们将详细讨论这些步骤。

**过滤**。我们首先将已知的良性和流行的域名与Alexa排名靠前的10000个域名进行比较[10]。其次，我们通过将输入域名与当前枚举的agd集合进行比较，过滤出已知的agd。收集基于DGA的恶意软件样本的一个好的起点是恶意软件分析报告和博客。使用这些作为来源，并根据我们的经验，我们确定了一组初步的22个家族DGAs。在其中的9个案例中，我们已经找到了DGA的重新实现，我们只需要验证。这使我们能够计算在DGA过滤器中使用的初始域集。

**识别**。在这一步中，我们将识别实现先前已知的DGA但使用新种子的示例，以及可能使用新DGA的示例。如果一个已知的域的输出与给定的dg表达式相匹配，我们就可以很快地确定它的输出。正则表达式捕捉主要特征，例如生成的部件的最小和最大长度（Lmin和Lmax）、DGA字母表，以及一组已知的tld。如果大多数输入域是由相同的模式匹配的，我们将样本标记为使用新种子的已知DGA。如果检测器产生不一致或不充分的结果，我们计算输入域的总体DGA分数，以测量这些域由DGA生成的可能性。这个分数是基于n-gram频率、熵和长度分析的。如果可以的话，我们会在NX域结果的情况下进一步提高分数。这种方法依赖于以前对DGAs的研究[13,18,34,43,54]。如果分数超过通过先前的实验和人工验证得出的阈值，我们将样本标记为使用新的未知DGA的候选。

**DGA枚举。**每当系统识别出一个潜在的新DGA时，我们就通过对相应的样本进行反向工程来手动验证这一点。如果存在，我们然后从二进制中提取域生成逻辑，重新实现它，识别使用的种子，最后计算由该算法生成的所有域。此外，通过分析该算法，我们更新了检测器集，以识别未来使用相同算法和不同种子的样本。

作为系统的输入，我们使用了一个特殊的沙盒feed，它由Shadowserver Foundation提供，由时间戳、恶意软件示例散列、DNS查询和DNS响应组成。此源包含新发现的恶意软件样本和正在重新处理的旧样本的混合，至少可以追溯到2009年。这种重新处理增加了遇到未知种子和可能不再活跃的家族的可能性。从该提要中获取2015年5月开始的3个月的数据，我们检查了总共1235443次沙盒运行，对959607个唯一域名执行了15660256次DNS查询。基于Shadowserver提要的数据，我们的系统使我们能够识别、分析和重新实现另外21个dga和表3中列出的大多数种子。

3.2重新实现DGAs

为了确保我们的重新实现是正确的，我们将其输出与在沙盒中执行时由相应恶意软件示例发出的域查询进行了比较。对于每个家庭来说，逆向工程和重新实现相应的DGA大约需要一天时间。在分析算法的基础上，我们从二进制中提取了DGA种子。在许多情况下，我们完全自动化了种子提取的过程，这使我们能够轻松地从同一家族的新二进制文件中提取种子。在解包恶意软件二进制文件之后，我们很少遇到原始解包代码的进一步保护层，这使得分析速度相当快。Nymaim和Suppobox的样本被严重混淆，并使用特定于家族的代码混淆。为了解决这些问题，我们分析了模糊处理技术，并实现了自定义的消隐器来自动消隐这些样本。Pykspa 2是一个异常困难的样本。与反向工程的种子推导函数不同，它更容易被仪器化，以便生成所有可能的种子值，我们随后在DGA的重新实现中使用这些值。

对系统先前识别出的样本进行人工分析，帮助我们消除了一些原本会是假阳性的样本。例如，在评估sandboxdns提要的过程中，我们遇到了许多示例，它们的DNS查询看起来像典型的AGD，但实际上是通过逆向工程验证的硬编码域。此外，我们只考虑DGA，其中AGD实际上被用作有效的潜在C&C端点，而不是像以下情况那样分散注意力。例如，我们发现Sality[1]的早期变体将动态生成的第三级部件前置到硬编码的域中，这是我们能够识别出的类似DGA行为的最早提示。另一个例子是宙斯衍生的城堡，当探测到虚拟环境时，它会产生未使用的诱饵agd。

我们知道，比本文所列的恶意软件家族更多的潜在使用DGAs。然而，我们相信，给定的数据集提供了足够的基础，以得出有意义的结论，对大多数影响的领域生成算法。

4 对DGA全貌的4个见解

在本节中，我们将全面概述43个DGA实现及其收集的253个种子。我们使用我们的分类学来描述和比较所研究的DGA。更具体地说，我们比较了域结构、有效期和生成方案。我们还研究了生成的域的随机性以及DGA作为C&C通信机制的优先级。表1概述了不同的DGA特性。该表包含具有不同DGA的某些族的不同变体。Gameover Zeus, Murofet, Pushdo, 和 Pykspa就是这样。

4.1域名结构

**字母表。**我们首先研究DGA用来生成域的字母表，用ü表示。字母表包含9到36个字符。虽然一些DGA故意使用短字母，但大多数带有小字母的DGA似乎是由有缺陷的实现造成的。第一种情况的一个例子是volatilecdar，它通过置换硬编码域的第二级部分来生成域(dotnetexplorer.net). Ramdo的DGA就是一个buggy实现的例子。在构建域名时，只选择索引为奇数的字母（即a，c，e，…），结果是只有13个字母的字母表。基于哈希的dga有16个字符的字母表，由a到f的数字和字母组成。

许多dga使用硬编码字符数组，通过迭代计算索引从中选择字符。这些dga中的一个常见错误是一个错误，其中一个或多个字符从未被选择。例如，Geodo省略了其字母表的最后一个字符，因此生成的域从不包含z。其他具有此错误的dga包括CryptoLocker、Necurs、Ramnit、Ranbyus、Shifu、TinyBanker和Torpig。他们都错过了字母表中的最后一个字符。对于使用多个独立数组从中选择字符的dga来说，这种错误更糟糕。例如，Mewsei使用两个数组，一个用于元音，一个用于辅音（有意或无意，缺少字母j）。一个off-by-one错误会导致每个数组中的一个字符丢失，从而使有效的字母表大小减少两个。Rovnix和Corebot使用两个独立的字母和数字数组，并遭受相同的故障，导致他们的AGD永远不会包含z或9。我们在所研究的DGA中发现了这个缺陷。

我们还观察到截断错误，其中字母表被截断成一个较小的数组，用于计算域。例如，Szribi将其26个硬编码字母截断为15个。在Urlzone中也发生了同样的错误，将可能的35个字符减少到只有32个。Torpig同时受到两种缺陷的困扰：Torpig的随机性是有缺陷的，34种可能性中只有30种是可以到达的（36减去2是由于“关闭1”错误造成的）。最大的字母表有36个符号，使用Dyre、Gameover DGA和Murofet 2。

**AGD长度**。生成的畴长为4～47个，中位数最小长度为9，中位数最大长度为16。14 DGA生产的AGD都具有相同的长度。在三种情况下（Banjori、Simda和Torpig），我们发现种子之间存在多个长度值，但一个种子的所有AGD都具有相同的长度。

**域名级别**。只有三个家族（Corebot、Kraken和Mewsei）使用一个或多个动态DNS提供者生成三级域。所有剩余的dga只生成一个二级域，然后与顶级域名部分连接。请注意，我们将构造视为co.uk或者com.tw为顶级域名部分，因为它们由单个注册表管理。Conficker和Necurs分别使用123个和43个TLD。这些案例清楚地说明了全球合作的必要性，以成功地摧毁一些僵尸网络。这也适用于使用较小TLD集的DGA僵尸网络。表2显示了所研究的DGA和鉴定的种子使用的最流行的TLD。这7个最常见的TLD在两个列表中是相同的，只是顺序不同。com和net是两个顶级TLD。我们观察到一种趋势，即使用流行的TLD而不使用区域参考，因为com和net一起占所有注册域的45%。我们假设攻击者希望他们生成的域名与良性流量很好地融合。

4.2域名有效期

超过一半的dga（24/43）是时间相关的，这意味着生成的域只在一定时间内有效。大多数dga（21/24）生成的域具有不相交的有效期。也就是说，在每个时间点，只有一组域是有效的。这三个例外是：1）Matsnu生成3个域，每个域连续3天有效，这意味着一次有9个潜在的C&C域；2）Pushdo，它将生成相对于给定日期的域，从过去的45天开始，到未来的15天。每天30个域名，一次可获得900个域名有效；3）Suppobox，每512秒产生一个AGD，然后在接下来的85个周期内有效，总计每AGD 12小时5分20秒。

11个依赖时间的DGA生成一天有效的域。其他族通过在日期上执行某些计算来提高域的有效性。例如，Gameover P2P和murofet2都将每月的某一天舍入到下一个最低值1、7、14、21或28，从而产生可变长度的有效期（1-7天）。Qakbot使用了一个类似的方案，但是使用了不同的四舍五入值（1、11和21）。Szribi的DGA[52]生成的域名有效期为1-3天。这种可变周期长度似乎是一个错误，因为DGA试图将一天四舍五入到最近的第三天（即Julian days）。

Pykspa的两个版本在生成域的有效性方面都有独特的特性。pykspa1每两天生成一个5000个域的列表，每个域的有效性取决于它在该列表中的位置。前20个域是随机生成的，而剩下的4980个域是从两个静态域集中选择的，每两天在这些域集之间交替。这是由于DGA将每个新计算域的初始32位种子提高到2的幂次，从而大大降低了计算域的随机性。为了提高检测的弹性，pykspa2生成的假域名的有效期比真域名短。当生成200个有效期为20天的真实域时，DGA生成800个假域，每个域的有效期仅为一天。

在所有依赖时间的dga中，Geodo在有效期方面具有独特的生成策略。它从对Microsoft网站的HTTP请求的响应中查询当前日期。然后，从硬编码的开始日期到当前日期，每隔900秒生成一个AGD。

4.3域名生成方案

基于算术的dga是迄今为止最常见的生成方案（37/43）。其中，26个DGA直接计算域中要使用的字符的ASCII码，11个DGA计算一个索引，用于从表示所用字母表的硬编码数组中选择字符。三个dga是基于词表的：Matsnu、Suppobox和Gozi。Matsnu和Suppobox将单词列表嵌入到相应的恶意软件二进制文件中。另一方面，Gozi从一个公开的文本文件中提取它的单词表，这在将来是不太可能改变的。例如，它使用美国独立宣言来生成诸如amongpeace knownlife.com网站. Matsnu和Gozi随机组合单词，直到达到一定长度，而Suppobox只合并384个单词中的2个，并添加.net作为TLD。

Bamital和Dyre这两个家族是基于散列的，使用散列函数MD5和SHA256在日期和域索引上的hexdigest输出作为输入参数。volatilecdar是唯一一个基于置换的家族，它可以产生170个可能的初始结构域置换。值得一提的是，Banjori、Tiny Banker、Urlzone和volatilecdar使用了一种域突变方案，其中种子或先前生成的域被用作计算后续域的输入。

4.4域名随机性

考虑到各种dga在其域生成方案中使用PRNGs，我们分析了生成域的随机性。为此，对于每个DGA，我们通过连接域的所有生成部分来计算全局字符串。然后我们计算这个弦的香农熵。这为算法的随机性提供了一个全局指标。最后，我们通过将计算值除以最大熵来计算相对熵，用Hrel表示。这使我们能够比较不同DGA的相对熵。为了简单起见，我们使用术语熵来表示Hrel。

我们提出以下意见。所有wordlist-dga都有一个显著较低的熵（Hrel<0.9），这是预期的，因为它们组合了字符不均匀分布的完整单词。然而，在基于hash的DGA-Dyre中观察到了最小的熵。虽然SHA256算法产生均匀分布的字符，但是Dyre在计算的hexdigest的[a-z]范围内预加了一个额外字符，从而破坏了均匀分布，因为hexdigest在[g-z]范围内不包含任何字母。在其他dga中观察到的低熵（Hrel<0.99）是特定实现选择的结果。我们在下面讨论这个问题。

1） 多个子字母表。通过将字母表拆分成多个不同的列表，并从这些列表中随机抽取字符以用于计算域名中的不同位置，某些dga会导致字符总体分布的不平衡。Bedep、Gameover DGA、Mewsei、Pushdo、Simda、Torpig和Virut就是这样。Redyms在每个AGD中使用连字符。

2） 不完美的PRNG。我们观察到导致不完全PRNGs的几个原因：首先，一些dga（Pykspa 1和Szribi）使用了自己设计的随机性不完全的prng。第二，其他dga（Murofet 1和TinyBanker）对所用prng的输出施加一定的条件，从而导致派生字符分布的不平衡。

3） 部分修改。一些dga根据初始域名计算下一个域。如果在初始域种子中没有引入足够的修改，这种变换可能会影响DGA的熵。此类别的一个例子是Banjori，它只修改给定硬编码种子域的前四个位置。

值得一提的是，一些DGA使用了众所周知的PRNG算法。值得注意的是，几乎三分之一的被研究的DGA（14/43）使用由递归关系Xn+1=（aXn+c）mod m定义的线性同余发生器（LCG）[36]，通过将使用的参数a、c和m与普通LCG的参数相匹配，我们发现10个DGA使用已知的LCG常量：4个DGA使用数值配方书[39]（NR LCG）中描述的实现，3个DGA使用Park and Miller[36]提出的最小标准实现（PM LCG），2个DGA使用Microsoft Visual C库（MS LCG），1个DGA使用Delphi LCG。其余4个LCG使用自选择常数。我们确定的著名PRNGs的其他用途是Mersenne Twister[33]和Xorshift[31]。五个DGA使用MD5初始化自己的PRNG，一个使用校验和CRC32。

4.5Command & Control Priority

超过半数的被研究的僵尸网络（23/43）使用其DGA作为唯一的C&C会合机制。此外，虽然其他5个家族（表1中用星标识）在转向其dga之前首先尝试硬编码域，dga也可以被认为是主要的通信机制。Banjori、TinyBanker和UrlZone首先尝试一个单独的硬编码域，该域随后用于计算相应DGA的初始种子。Matsnu包含多个优先的硬编码域。然而，同样的域也会在某个时间点由其DGA生成。Nymaim包含一个遗留的主要硬编码域，这个域长期以来被削弱，可能被botmasters忽视，这意味着它主要依赖于它的DGA。

其他家庭在其主要通信方法（通常是硬编码域）失败时使用其dga作为备份机制。例如，Rovnix试图连接到它的一个硬编码域。如果这失败了，它会求助于它的DGA，然后最终尝试通过不可见的Internet项目（I2P）进行连接。其余的家庭使用他们的dga作为最后的手段来联系C&C服务器。

这清楚地表明了dga在现代僵尸网络中的流行。因此，我们认为，不应将DGA主要视为备份机制，而应视为主要的C&C概念。

5 DGA域使用

在本节中，我们将介绍我们对DGA在实践中如何实际使用的分析结果。我们以过去14年的WHOIS数据为基础，对家族活动周期、DGA域的注册状态和botmaster注册策略进行了深入分析。我们还分析了缓解响应时间。

5.1 WHOIS数据集

我们的分析基于DomainTools WHOIS数据集，其中包含过去14年收集的超过90亿条WHOIS记录[19]。基于我们的重新实现和可用的种子，我们计算了所有所研究的43个DGA可能的域名。我们向DomainTools提供了计算出的域名，他们友好地向我们提供了他们在数据集中找到的DGA域的WHOIS记录。DomainTools为本研究提供的数据于2015年9月22日编制。这使我们能够识别出115387个域名的303165个与DGA相关的WHOIS记录。

必须特别注意依赖时间的dga，它们使用日期信息来计算其域。为了确保良好的覆盖率并尽量减少我们漏掉的DGA域集，我们计算了在任何公开的起始日期之前一年到2015年12月31日（在DomainTools数据集中最后一个WHOIS记录后三个月）覆盖的所有域。这个结束日期提供了大约3个月的前瞻性，检测在实际使用之前注册了一段时间的域。

这些WHOIS记录包含以下字段

•WHOIS记录日期

•WHOIS记录更新日期

•域创建和到期日期

•注册人姓名

•注册人姓名和电子邮件地址

•在WHOIS中注册的名称服务器

这使我们能够推断出以下信息：

1） 域名第一次注册。这是基于WHOIS记录的日期和域创建的日期。

2） 僵尸网络活动的开始。任何域的第一个记录隐式地指示域已注册，并帮助估计相应的恶意软件系列何时开始其操作。

3） 域所有权的变化。WHOIS记录的更新日期通常与域的责任变更相对应。可以通过对registrator、registrator和registered nameserver字段进行相应更改来验证这一点。这使我们能够通过观察WHOIS记录从非深坑操作员到深坑操作员的变化，得出潜在C&C领域何时得到缓解（例如，深坑）。我们确定了29个不同的组织运行坑，但我们不会透露更多的细节，用于检测坑组织，以保护他们的运作。

4） 域名parking

registrator、registrator和registered nameserver字段传递信息，我们使用这些信息来标识域名parking，以便调查如何为获取意外流量或转售而持有通用AGD。总共，我们确定了36个服务提供域名parking或域名转售。

我们的数据集涵盖了除五个以外的所有研究家族：Conficker和Virut生成了大量的域，因此没有包含在提供给我们的数据集中。Corebot、Kraken和Mewsei

使用基于DDNS的DGA，因此在原始的DomainTools WHOIS数据集中不可用，后者只包含主域名而不包含子域。值得注意的是，Bedep所使用的非确定性播种并不能阻止追溯分析，因为我们仍然能够通过收集一段时间内使用的汇率来生成所有域。

5.2家族活动周期

在这一节中，我们将分析所研究的僵尸网络处于活动状态以及为恶意目的注册域的时间段。为此，我们首先确定数据集中第一个相关记录的日期，用T-first表示。在许多情况下，这是第一次观察到的整体登记。在数据表明与良性区域发生碰撞的情况下（cp.第5.4节），我们将反向分析第一个坑事件的记录。然后，我们确定域被取下或最后注册的时间，用T-last表示。所研究的僵尸网络的详细活动周期如表3所示。对于我们没有数据的家庭，他们各自的T-first是根据前面提到的公共来源估算的。

在Bamital, CryptoLocker, and Gameover P2P,案例中，tlast表示被封锁的日期。对于Conficker，Tlast标识最后一个已知Conficker变体从受损系统中删除的日期[26]。一个特殊的例子是Rovnix，我们只识别了一个由一个地坑运营商进行的注册，因此决定让Tlast保持开放。

在确定了每个系列的开始日期（Tf irst）之后，我们删除了之前计算的agd。这使得计算集总共减少到165161439个AGD，其中159712234个在所有DGA中是唯一的。此外，我们删除了在DomainTools数据集中不可用的族的计算域，留下一组18446125个唯一的agd。我们用Duniq-1表示这个集合，并将其用于以下所有检查。

对Tf-irst的进一步研究表明，尽管这一概念在2007年首次使用，但超过一半的分析DGAs（25/43）是在2013年及以后引入的。似乎恶意软件作者等待早期采用者收集运行基于DGA的僵尸网络的经验，然后再自己尝试。活动时间短的案例表明，一些作者似乎也只试验了DGAs。例如，早期版本的Geodo使用硬编码的IP地址进行C&C通信，而一个带有DGA的版本只使用了大约6周。之后，botmasters又回到了原来的通信方式。

Murofet2只活跃了三个月，之后它广泛的后续产品gameoverp2p（使用了非常相似的DGA）出现。Gameover DGA，另一个在Gameover P2P被收购后不久出现的继任者，只活跃了大约5个月。尽管二进制差异意味着它依赖于与Gameover P2P相同的源代码，但还不确定它是由同一位原作者Slavik[42]操作的，还是仅仅用作分散注意力的手段。

在撰写本文时，我们使用Tlast来识别活跃家庭。如果某个族的DGA生成了一个在WHOIS数据集覆盖的时间段的最后一个月内使用的域，则认为该家族是活动的。在此基础上，我们确定了10个家族：Gozi、Nymaim、Suppobox、UrlZone、QakBot、Matsnu、Dyre、TinyBanker、Bedep和Shifu。这是估计活动族的一个下限，因为有些域名家族不使用DGA作为其主要的C&C会合机制。

5.3域名注册状态

在18446125个唯一生成的AGD中，实际注册了115079个（0.62%）。其中72,37%的注册AGD对应于sinkhole操作者，而其余的27,63%为non-takedown域名。我们将注册的AGD分为四类：

1） 预注册域名（30.25%）。在Tf irst之前已注册的域。这些域通常是在相应的僵尸网络出现之前注册的，而且很可能是良性的。

2） 缓解域（3.76%）。最初由non-singhole注册人持有，但后来改为inskhole运营商的领域，通常表明该领域已被缓解（第5.6节）。

3） 纯sinkhole域（16.24%）。从一开始就由sinkhole操作员注册的域。

4） 剩余域名（49.75%）。此集合包含在Tf irst和Tlast之间注册的域，但我们无法可靠地将它们标识为sinkhole，或以其他方式判断它们是良性的还是恶意的（主要是由于使用了WHOIS隐私服务）。然而，良性域的数量（至少对于非单词表dga）极有可能是少数，因为少数dga具有预先注册的域，并且大多数dga都具有共同的随机性和域长度。

5.4域冲突

为了评估AGDs是否是可靠识别僵尸网络的良好特征，我们分析了不同dga生成的域之间的冲突。在43个DGAs中，两个攻击性的DGAs Virut和Conficker相互之间以及与Necurs、Nymaim和Pykspa 2发生了碰撞。更具体地说，Virut与Pykspa 2发生了1791次碰撞，Conficker发生了1316次碰撞，与Nymaim发生了179次碰撞。Conficker与Pykspa 2有11次碰撞，4次与Necurs碰撞，1次与Nymaim碰撞。所有的碰撞域都有5-6个字符长。除此之外，只有Nymaim和Pykspa 2有一次碰撞，因为它们都产生了AGD wttttf.net网站.

接下来，我们分析了DGA生成的域名与良性域名之间的碰撞。为此，我们使用Alexa列表[10]，并将其与每个DGA生成的域集进行比较。在这里，只发现了少量的碰撞。Virut与Alexa top百万域有2507次冲突，相当于其唯一AGD的0016%。对于所有其他的dga，我们只发现了27个与Alexa列表的冲突：24个与基于wordlist的dga的冲突（Suppobox:21，Matsnu:2，Gozi:1），以及3个与6个字符长的Pykspa 2域的冲突。

然后我们研究了DGAgenerated域和在相应的僵尸网络出现之前（Tf-irst之前）已经注册的域之间的冲突。正如人们所料，基于单词表的dga所观察到的冲突数量最高：74.39%的supbox agd和40.0%的Matsnu agd与已有的域发生冲突。尽管Gozi是一个wordlist DGA，但它与已经注册的域的冲突很少。这是因为DGA会以33%的概率截断一个选中的单词，然后将其附加到计算的AGD中。这将创建许多包含断开单词的域，使得意外注册的可能性大大降低。对于其余的家族，928个（92.24%）预先登记的AGD中有856个（92.24%）的长度为5到6，而更长的都是一个意外的单词（例如。退伍军人.kz)或者很容易发音（例如。kankana.com网站, kandilmed.com网站).

这些结果清楚地表明，不基于单词表的dga和生成长度超过6个字符的域名（34/43dgas就是这种情况）是检测相应僵尸网络家族的可靠来源。

5.5域名注册

时间依赖DGAs的AGDs的一个特殊性质是，它们只在一定时间内有效。在本节中，我们将深入了解依赖时间的DGA的域注册前瞻性，即从注册域到其生效的时间点（由DGA在当时生成）的时间量。在这里，我们区分了sinkhole和non-sinkhole注册（可能由botmasters注册）。

图2提供了数据集中22个与时间相关的DGA的结果概述。对于每个族，使用箱线图对结果进行总结，该箱线图显示了DGA域相对于相应有效期（第零天）开始的注册时间。也就是说，x=-1处的注册事件意味着相应的域在其生效前一天被注册。为了获得更好的可视化效果，我们选择了对称的混合线性（前10天）和对数（所有超出的）比例。这样可以在大多数有效期内以更精细的分辨率显示数据，但也允许包含最多一千天之外的事件。为了更好的定位，红色条表示每个家庭的AGD的整个有效期。在这张图中，我们对每个家庭区分了四种可能的类别：1）non-sinkhole注册；2）sinkhole（用S标记）登记；3）pre-takedown登记；4）post-takedown登记（用†标记）。请注意，注册号可能与表3中的不同。这是在Tf irst之后绘制关于有效期开始而不是注册日期的数据的结果。

主要观察到的22个DGA中，有14个sinkhole操作员non-sinhole操作员更早地注册了这些域名。比较这些的中位数发现，7个例子的sinkhole注册发生在1～10天之间，而其余7例则发生在18～143天之间。143天适用于Torpig，在那里，大多数的sinkhole域名都是提前5个月注册的，有效期很短，这似乎是一种尝试。第二大差异（48天）是CryptoLocker。对于3/22dgas，sinkhole和non-sinkhole同时注册。在剩下的5/22 DGA中，non-sinkhole注册发生在sinkhole注册前2至37天。

在Gootkit的例子中，一个sinkhole操作者在DGA域生效的同一天注册了它们，这表明对这些域的反应性响应被恶意软件使用，但尚未被采取，这可能会阻止botmaster注册它们。

Sinkhole操作者通常不担心他们的域名会被其他实体抢走，这就解释了sinkhole玩家在botmaster之前注册域名的普遍趋势。此外，这使他们能够在这些域变得有效之前评估这些域的传入流量。另一方面，botmasters必须假设他们的域最终会被缓解。因此，它们在有效期之前或之内注册自己的域，以确保当机器人试图联系它们时这些域的可用性。这意味着，持续监测有效期在取下日期前后的AGD是成功的必要条件。

对于这三个家族来说，地坑登记至少发生在AGD有效期之前的几周甚至几个月。这就保证了僵尸主机不能通过注册更多的agd来快速恢复对僵尸网络的控制。虽然Gameover P2P的agd的封锁似乎已经完成，但Bamital和CryptoLocker的agd都是在封锁后由non-sinkhole操作者注册的。在CryptoLocker的情况下，这些域在其有效性开始前不久就被注册，这表明并不是所有域都通过同一个中心机构被阻止。

我们在Bamital的案例中观察到了一个有趣的模式。2011年1月1日，同一注册人注册了2012年1月4日、2013年1月3日和2014年1月4日的AGD域。我们认为，Bamital的操作者这样做是为了及时确保保险域的安全，以备日后被摧毁时使用，并推测这些域不会被发现。我们发现Nymaim 和Murofet.的botmaster使用了类似的策略。如果是Nymaim，则在2012年12月6日和2012年12月9日，

注册了两个分别在1年和2年内有效的域名。对于Murofet，在2011年3月12日和随后的3天内，4个域名在1年内有效，5个在2年内有效，3个在3年内有效，3个在4年内有效。他们都使用不同的，显然是假的身份，但注册人是同一个。这表明，在准备takedown活动期间，应提前相当长的时间检查AGD的潜在注册情况。

Pykspa 1号和Suppobox的sinkhole注册数量少得惊人。对于这些家族来说，大多数AGD都是由单独的sinkhole操作者在较长时间内以相同的前瞻性进行注册的。sinkhole操作者只注册一小部分可用域表明僵尸网络监控操作。另一方面，我们观察到许多不同的实体参与了Suppobox领域名的non-sinkhole注册。考虑到与良性结构域相对较高的碰撞率（第5.4节），我们认为其中许多域属于良性注册者。这表明suppoboxdga与合法的域所有者很好地融合在一起，允许botmaster将其C&C隐藏在良性域中。

我们观察到一些关于域名注册的特殊性。对于sinkholers和其他人来说，Gozi的注册往往都会迟到。对此我们没有找到一致的解释。另一个奇怪的现象是斯里比的sinkhole登记。2008年僵尸网络活跃时有8次注册，2015年又有32次注册，比僵尸网络消失4年多。此外，这32个域名都是在失效一个月后注册的。我们认为，各自的sinkhole组织使用了错误的DGA重新实施来计算AGD。

5.6缓解响应时间

我们所说的“缓解Mitigation”是指将域名所有权从一个non-sinkhole转变为sinkhole。在本节中，我们将分析这些事件的时间偏移量，以衡量这些操作的有效性。在这里，我们只考虑DGA，其中确定了10个或更多与收购无关的缓解措施。表4总结了这次检查的结果。在大多数情况下，针对新DGA或种子的第一次缓解在一周内进行。

人们可以预期，在初步识别出DGA或DGA的新种子之后，所有进一步的缓解措施都将具有更短的响应时间，因为应该知道潜在生成的域。我们观察到的中位反应时间很短。这些也是DGA，其中大多数AGD在其各自的有效期内得到缓解。另一方面，针对Bamital和CryptoLocker的对策非常无效，因为它们主要针对不再有效的AGD。

第一次缓解后，所有其他DGA的响应时间增加。有趣的是，在同一天对这些DGA进行了多次后续缓解。这种模式对应于辩护人的一种常见做法。在识别初始恶意域之后，他们通过执行反向IP查找来查看指向同一个C&C服务器的其他域。这表明，在最初确定的AGD之前，可能已经注册了其他AGD。在Nymaim、Ramnit和UrlZone中观察到了这一点。

5.7 DGAs和域名parking

接下来，我们将检查有多少AGD注册到域名parking中。对于这些域的识别，我们使用了与Vissers等人类似的程序。[51]。我们总共找到6458个AGD，其中包含一个在某个时间点指向parking的登记。这对应于通常的做法，即在原始所有者的注册期结束后，过期的域将被转移到域parking或由域转售商取回。

令我们惊讶的是，对于3852个AGD来说，parking注册也是第一次，而且在许多情况下只是注册条目。特别是，第一次parking注册事件构成了以下DGA登记的重要部分：Banjori 620（90.78%）、QakBot 595（54.69%）、Pykspa 2 883（45.82%）、Necurs 122（41.36%）、Pushdo TID 91（37.14%）、Ramnit 286（30.46%）、Murofet 1736（23.20%）。除Pykspa 2外，这些DGA都没有大量的预注册| RP |，这使得意外或意外注册极不可能。同样值得注意的是，对于这些dga，有2917个（87.52%）域在同一个域parking服务处注册。关于有效期，2336份（19.86%）时间依赖型AGD中，464份是在各自有效期结束前登记的。在这种特殊情况下，与时间无关的agd一起，至少1461个agd可能被滥用，以驱动来自受损主机的自动生成的流量流向域parking系统。

5.8讨论：对抗DGA

在本节中，我们总结了前几节中的观察结果，并讨论了它们如何影响针对dga的对策。

从域生成方案的分布来看（第4.3节），43个DGA中只有3个基于单词表，因此产生了一些有意义的域名。另一方面，对检测随机生成的域的方法进行了充分的研究[13，34，54]，我们的数据表明，这些方法在未来仍具有相关性和适用性。

正如对活动期的研究所示（第5.2节），DGA与恶意软件作者的关系非常密切，尤其是在过去两年中，43个被认为的DGA中有25个在2013年及以后浮出水面。

DGAs的核心概念之一是隐含的经济不对称性：一个有效的域授予攻击者控制权，而防御者需要拒绝对所有潜在域的访问。实际注册的agd的总体数量较低（第5.3节），这突出表明，攻击者在操作僵尸网络时只需要很少地使用所有潜在的域。此外，由于大多数攻击者似乎在域的有效性或使用前不久就注册了域（第5.5节），因此它们很有可能被突然袭击，并且可以取消dga的备份实用程序。然而，提前4年确认注册事件意味着未来的域名应该被仔细检查，以确保成功删除

我们的数据表明，在过去的takedown中，域名都是大规模takedown的，与takedown相关的资金投入非常大。由于一次被takedown而注册的域允许它们被用作sinkhole，以便在被破坏的系统上收集遥测数据。然而，每个有效期内的单个AGD足以达到相同的监视效果，类似于攻击者只需要一个域，只要剩余的域不可用。因此，我们建议提高对DGAs与ICANN的相关性和内在机制的认识，并在注册运营商层面广泛使用阻塞agd。对于大多数AGD，阻塞的预期影响实际上是可以忽略的：所分析的DGA中有34/43的AGD的最小长度为7或更大，对于这些AGD，我们观察到其整个AGD集基本上没有与现有域发生冲突（第5.4节），这意味着这些域似乎不太可能出于良性目的进行注册。在依赖时间的DGA的情况下，一旦各个AGD的有效期已经过去，这个阻塞甚至可以被解除，这可能是对基于地址词表的AGD的折衷。

此外，我们在159712234个独特的DGA域内发现了3302个DGA之间的碰撞（第5.4节）。这意味着像我们的数据集这样的agd查找数据库可以作为一个非常可靠的资源来帮助基于基本上没有误报的接触域来识别恶意软件家族。作为对社区的服务，我们继续收集有关DGAs的信息，并通过dgaarchive免费提供这些数据[38]。

6相关工作

其中，以下工作对域生成算法进行了详细的研究。Stone Gross等人。[50]曾在2009年初对Torpig进行过僵尸网络收购。基于逆向工程，他们利用了这个事实：这个家族使用的是一个域生成算法，而没有在C&C协议中进一步的保护机制。他们的出版物是学术文献中最早详细阐述DGAs概念的文献之一。Barabosch等人[16]根据时间相关性和因果关系这两个特征定义了DGA类型的分类法。他们还解释了一种自动定位DGA相关代码的方法，使用动态分析进行解包和API跟踪，并结合数据流分析进行代码提取。Mowbray等人。[34]已使用收集的DNS数据识别潜在的DGA域，检查查询源IP地址和查询域名的长度分布。通过这种方法，他们确定了19种不同的AGD方案，并列出了表1中给出的特征子集。

有几项工作是基于收集的网络流量来检测DGA和其他恶意使用的域。2010年，Yadav等人。[54]对域名中的字符分布和ngram进行了评估，以便通过异常检测生成的域名。Antonakakis等人。[13] 介绍了Pleiades，一个基于DNS的系统，通过监视不成功的DNS请求来检测潜在的DGA域簇。正如我们的工作所证实的，他们建立他们的系统的前提是只有一小部分AGD是实际注册的。他们通过评估一家大型ISP的流量，找到了6个当时未知的新DGA。Schiavoni等人提出了另一种使用DNS数据Phoenix的方法。[43]。他们定义了一个可发音域的模型，然后检测出偏离该模型的那些域。他们的系统还对识别的域进行分组，并允许跟踪基于DGA的僵尸网络的活动。Bilge等人。[18] 介绍了他们的系统暴露。它用于检测恶意域名的基础上选择15个特征，可观察到DNS流量，包括基于时间，基于DNS应答，TTL和域名特征。

还有一些作品探索动态和主动的黑名单功能。2009年，Ma等人。[28]比较了不同信息源对黑名单的有用性。他们发现WHOIS的数据，特别是登记的时间信息，在这方面非常有价值。Antonakakis等人。[12] 已经提出了Notos，一个自动给域名分配分数的系统，可以用来自动生成黑名单。它们关注域字符串和tld的特性。徐等。[53]提到了预先生成AGD以启用预测阻塞的想法。然而，他们并没有评估这一想法的有效性，正如我们在本文中所展示的那样。

最近，Vissers等人。[51]利用DNS和WHOIS数据集，研究了域名parking服务与恶意域名的关系。Nadji等人。[35]已经开发了一个有效的僵尸网络takedown的概念，在这个概念中，他们确定覆盖潜在的DGA存在是提高效率的重要步骤。关于C&C交会机制进一步创新的分析，Holz等人。[23]研究了快速流量服务网络，而Rossow等人。[40]对使用P2P机制的僵尸网络进行了调查。

7结论

在这项工作中，我们首次提出了现代僵尸网络所使用的域生成算法的全面测量研究。我们的研究基于对43个恶意软件家族和变种的dga进行逆向工程。使用这些算法的重新实现，我们生成了159712234个唯一的DGA域的集合。然后，我们利用DomainTools提供的历史WHOIS数据对域名注册进行了分析。我们的主要发现是，我们的域数据集既可以用于预测性阻止尝试的C&C访问，也可以准确地确定恶意软件家族和活动，基本上没有误报。此外，我们描述了botmasters和sinkholers的注册行为，并检验了域名缓解的有效性。

作为进一步的贡献，我们不断收集关于DGA的更多信息。我们的工作产生的完整的域数据集在这里免费发布：https://dgarchive.caad.fkie.fraunhofer.de.。此名为DGArchive的web服务提供反向域查找以支持恶意软件分析，以及前向生成域列表，这些域列表特别可用作网络保护的阻止列表。