Office of Liquor and Gaming Regulation

Decentralised Draw Systems

Version 1.0

Queensland Government

For further information, please contact the Office of Liquor and Gaming Regulation on 13 QGOV (13 74 68) or visit https://www.business.qld.gov.au/industry/liquor-gaming

1 Contents

2 Introduction

This document is released for discussion and concept development purposes only and does not reflect the official position of the Office of Liquor and Gaming Regulation (OLGR) or the Queensland government. This document is not an OLGR technical requirements document.

This document describes a new type of secure, draw system called a **Decentralised draw system (DDS)**. A DDS potentially offers increased security for draw devices and random number generators (RNGs) in general. Decentralised draw systems can be used to increase draw security in many applications requiring random numbers while in turn decreasing setup and operating costs for any given level of operating security.

A DDS potentially:

- Facilitates extremely secure, hard-to-defeat draw systems and devices.
- Facilitates fully automated draw systems (including auditing).
- Alleviates the requirement for third party physical tamper sealing of RNGs / draw devices and the associated requirement to perform seal inspections thereof.
- Alleviates the requirement for third party build verifications of most RNGs / draw devices.
- Alleviates the requirement to make RNGs a discrete physical device[i]. In other words, a draw device / RNG in a DDS can potentially be incorporated directly into the hosts they serve.
- Allows two or more parties to easily perform a random draw and be able to implicitly trust the integrity of draw result.

A decentralised draw system may be used for a wide range of applications requiring securely generated random numbers such as lottery systems and gaming systems in general. A DDS is suitable for use as an automated (computer driven) draw system, or as a manual (human driven) draw system.

A DDS is comprised of two or more draw participants (DPs). Refer to section 4.2 for more information on draw participants.

The most notable advantage of a decentralised draw system is it can achieve **an uncapped, scalable level of security** in random number generation in order to meet any given level of operating risk. The level of security in a DDS is generally proportional to the number of draw participants in the draw.

Refer to section 4.4 for a full list of benefits pertaining to a DDS.

Another notable feature of decentralised draw systems is that they can allow groups of **remotely located** people, computers, organisations, countries, etc., to securely generate random numbers for any given purpose. The draw participants do **not** have to establish any trust in each other for a draw outcome to be fully secure. Each draw participant need only trust the RNG/entropy they bring to the draw for any draw outcome to remain fully secure. This has applications outside of gaming, refer section 6.3.

---

[i] In the gaming industry, RNGs are typically discrete devices for security reasons.

A DDS with a sufficient number of draw participants is highly tolerant to attack. For a DDS to be successfully attacked, the attacker must successfully attack every draw participant.

The following definitions apply in this document:

> *"**Draw**" refers to the process of generating a set of random results for use with any application, game, decision, or any other purpose that requires random numbers.*
>
> *"**Draw system**" refers to an arrangement of hardware, software, procedure, or process, or any combination thereof, whose purpose is to generate random numbers for use with any application, game, decision, or any other purpose that requires random numbers.*
>
> ***"Draw participant"** means a machine, computer, software, or procedure (human driven or automated) or any combination thereof which can independently generate random numbers for their participation in a draw within a DDS. Refer section 4.2 for more information about DDS draw participants.*
>
> *"**Draw device**" refers to an RNG device or a process that provides random numbers.*
>
> *"**Decentralised draw system**" (DDS), is a draw system comprised of multiple draw participants, arranged so that they can communicate with each other. To conduct a draw each draw participant creates entropy, sufficient for the draw alone, which is then fairly and securely combined via a specific procedure (defined in this document) to produce the final draw result. A DDS may also be referred to as a **Distributed Draw System**.*
>
> ***"Entropy file"** is a collection of random bits generated by one or more draw participants for use in a DDS draw. Typically a computer "file". Any single entropy file created a single draw participant contains sufficient random numbers to conduct the intended draw.*
>
> ***"Fingerprint"** in the context of a file or data denotes the output of a secure hash algorithm applied to the file or data. Refer http://en.wikipedia.org/wiki/Secure_Hash_Algorithm*

The main benefit of decentralised draw systems is that they enable significant increases in the security and integrity of draw systems. Decentralised draw systems permit fully automated draws to be conducted at never before achieved levels of security. Decentralised draw systems have benefits in almost any type of application that require transparent, secure random outcomes.

There are two primary modes of operation envisaged for decentralised draw systems: automatic and manual. A decentralised draw system which is fully automated is typical for high frequency use, and a decentralised draw system which is predominantly manually driven (e.g. by human draw participants) is most suitable for low setup cost and low frequency / emergent / ad-hoc use such as a disaster recovery draw system. Note that even with human draw participants, each draw participant will generally require access to a computer due to the requirement for the use of an RNG, a Secure Hashing Algorithm and XOR operations in the DDS draw procedure.

Each draw participant in a DDS need only be primarily concerned with their own integrity and security. In a DDS, provided that at least one draw participant and their draw device / RNG is good (i.e. secure and random), then the overall draw outcome will also be good.

Scope

The following items are outside the current scope of this document:

- Methods to ensure that the final draw result produced by a decentralised draw system is actually used, imported into a host system, and correctly published.
- Methods to ensure winners are notified and paid.

Draw participants would invariably have a role in the implementation of the above items.

2.1 Abbreviations

PRNG	Pseudo Random Number Generator
RNG	Random Number Generator
SHA	Secure Hash Algorithm
XOR	Exclusive or (logical)

3 The DDS draw process

This section demonstrates the typical draw process of a decentralised draw system. Implementation of an automated draw system using computer based devices will be evident once the draw process is understood.

3.1 One time draw preparation

One-time draw preparation items occur typically once per type of draw / game type. The one-time preparation items are as follows:

- Determine who the draw participants will be and number thereof. For more information about draw participants, refer to section 4.2.

- Based on the specifics of the draw / game, determine a suitable result scaling algorithm and the amount of entropy required in order to conduct each draw.

  - *In most cases a simple scaling algorithm can be used, e.g.* `Result = R mod RANGE` *(where R is the random number) and a corresponding exact amount of entropy. However, in some cases (e.g. where the required RANGE is large when compared to the range of R), a zero bias scaling algorithm should be used which requires the amount of entropy required for each draw to be somewhat more than is necessary*[ii]*. This is because zero bias scaling algorithms occasionally need to discard some bits of entropy.*
  - *Minimum entropy requirement: The amount of entropy generated* <u>*must*</u> *be at least the size of the output of the SHA / fingerprint being used in the DDS draw process. This is to prevent any brute force attacks by any draw participants e.g. even if the draw is just a coin toss (1 bit), if using SHA-256, then a minimum of 256 bits of entropy from each draw participant should be generated.*

- Each draw participant must acquire a cryptographically secure RNG whose entropy output is predominantly hardware based. Each draw participant's RNG must be capable of conducting the draw alone and preferably in sufficient time in order to avoid unwanted delays *e.g. a pick of 3217 tickets from 225,000 tickets requires approximately 58,000 bits of entropy; equivalent to only a 7.2kb file but a big hit in one go for most RNGs. Thus the entropy bit rate may be a factor in RNG choice for some applications.*

- Draw participants may previously have confirmed the identity of each other to a degree commensurate to the operating risk of the draw (e.g. meeting in person, exchanging public keys or use of certificates). However the need to authenticate fellow draw participants is somewhat diminished in a decentralised draw system depending on the specific application. In a decentralised draw system you can have compromised draw participants and the draw can still take place and the result be

[ii] How much additional bits of entropy to be safe is the question? It is not acceptable to run out of entropy during a DDS draw process. The answer can only be calculated within a desired confidence interval once all scaling requirements for the application are known. The subject of RNG number scaling is well understood and is generally outside the scope of this document.

100% secure and random provided at least one draw participant was not compromised.

- A DDS ***draw parser*** must be created, shared, verified and agreed upon by all draw participants.

  The draw parser is typically represented via a simple program written in a high level scripting language for a platform independent interpreter. Alternatively it could also be a represented by a procedure in a manual draw system for very simple games.

  The *draw parser's* job is to take as input, the final XOR combined entropy bits and convert them into a human/computer readable draw results file (e.g. formats: txt, csv, ini, json, XML, Python, Lua, etc.). In other words, result scaling and formatting according to the rules of the game/application the draw was for.

  The draw parser must be agreed on and shared among all draw participants and its functionality verified in full by each. For this reason, the draw parser should be in an easily understood / verifiable format. High level scripting languages are advantageous here as they are simple to write, understand, are system independent, and are easily shared in source form.

- Each draw participant authenticates and prepares their draw device. Typically, for manually conducted draws, this will be in the form of one or more applications running on a computer (comprised of an RNG, a SHA tool, an XOR utility and a secure communications suite).

- Establish a communications methodology for the draw participants to communicate during a draw.

  To expedite the draw process, draw participants should use a reliable electronic means of communication (i.e. the ability to quickly exchange small text messages or files). In manual draws conducted by human draw participants, ad-hoc communication methods may include an on-line chat program or similar application. To avoid binary file exchange (re the entropy bits) if desired, entropy bits could be converted to a hexstring or base64 to allow easy exchange over text based communication methods.

3.2 Draw procedure – short summary

1. **Make contact.**
    a. Draw participants make contact and perform introductions.
    b. Draw details / parameters are be confirmed and agreed.
2. **Create entropy bits.**
   Each draw participant creates entropy bits sufficient for the draw (as if they were conducting the draw alone) <u>and keeps it secret</u>.
   This individual entropy is called the ***entropy file*** throughout this document.
3. **Share entropy file fingerprints.**
   Each draw participant creates and shares a cryptographically secure fingerprint of the *entropy file* with all other draw participants (e.g. SHA-256 or better).
4. **Share entropy files.**
   Once all entropy file fingerprints have been shared across all draw participants, draw participants can now share their *entropy file* with all other draw participants.
5. **Finalise draw.**
    a. Draw participants verify that the *entropy file* fingerprints match the respective *entropy file*.
    b. If ok, each draw participant creates the final combined draw *entropy file* (via XOR).
    c. Run the final combined draw *entropy file* through the agreed draw parser to produce the final human/computer readable draw result.
    d. Confirmation of the final draw result (or fingerprints thereof) between the draw participants (i.e. they should all match).
    e. Draw complete. Publish / import / sign off on draw.

3.3 Draw procedure - in full

- Draw participants make contact on the agreed communications medium and perform the necessary introductions and identify verification as desired.

- Draw participants confirm the intended draw details / parameters.
    - If the amount of required bits of entropy varies from draw to draw for the given application, then the draw participants may need to mutually confirm the amount of bits of entropy required for each draw. Often the amount of entropy bits required can be a simple deterministic process and is easily derived from the draw details. In this case, draw participants need to agree on a simple algorithm. Typically it amounts to a mutual agreement on the size of each random number pulled out of the raw entropy (which would be number divisible by a power of 2 and large enough with respect to the draw details[iii]).
    - A deterministic method of creating a unique identifier for the draw (or draw uid), based on the draw parameters is recommended. Example draw uid components are (use depending on risk):
        - A unique game/product name.
        - Date and time stamp.
        - A draw serial number.

[iii] 32 bit or 64 bit numbers will commonly be used.

        - A SHA of applicable bets. (This can help prevent attacks on bet databases if applicable).
        - A rolling SHA hash.[iv]

- If all concur with the draw details / parameters, each participant **creates** an *entropy file* sufficient to conduct a draw alone using their RNG. The file format may be binary, or any format capable of representing binary data (such as base64). **Draw participants must keep this file confidential at this stage.**

- Each draw participant generates a **fingerprint** of the *entropy file* (a secure hash value of sufficient integrity such as a SHA-256 hash) and shares just the fingerprint with all the other draw participants. **DRAW PARTICIPANTS MUST NOT SHARE THEIR *ENTROPY FILE* YET**.

- Once all entropy file fingerprints have been successfully **shared** between draw participants, the draw participant's *entropy file*s are now **shared*** among all the draw participants ***ONLY NOW AND NOTBEFORE**.

- Each draw participant **verifies** the received files against the respective previously received *entropy file* fingerprints for all draw participants.

  On success, each draw participant **combines** all the received entropy files into a single final *entropy file*. This is performed typically via a simple bit XOR merge utility. On any failure, an attempt may be made at addressing the problem by sourcing the correct *entropy file* for the previously submitted fingerprint (but not the other way around as this would be a security risk). If all problems can be corrected in this way, then the draw can continue. If not, the draw must **not** continue and the reason for the issue should be investigated.

- Each draw participant applies the DDS *draw parser* to the now combined single entropy file, creating a final human/computer readable set of draw results. This should result in each draw participant having identical final draw results.

- The fingerprint of the final draw result may be generated by each participant and verified against the final fingerprint of all other participants. All participants should end up with the same fingerprint.
    - If not, the draw has critically failed and must not proceed. This should never happen if due diligence in draw preparation has occurred. If it's not a draw participant causing the issue, then it's a mistake in draw one-time preparation, such as not choosing a platform independent language and interpreter for the draw parser.
    - If the draw result fingerprints all concur, then the draw is a success and can proceed as per below.

- Each draw participant should sign the final draw file and share copies of the signatures with other draw participants and publicly publish as applicable.

[iv] A rolling SHA hash running over previous draw results can be used by draw participants to ensure that they have not missed or been excluded from any draw.

- The draw results can now be accepted and the draw is essentially complete. The results may then be used for their intended purpose, for example imported into a host gaming system, published and winners determined.

It is generally recommended that each draw participant:

- Confirms that the correct final draw result is actually the one used by the target system / product. This can be accomplished via digital signatures, a read back or an audit of published results.

- Publically publishes their draw results individually in an authenticated manner (e.g. https). This promotes draw transparency and allows anyone to verify a draw at their discretion with any of the draw participants.

Notes:

- How the final results make their way to their intended destination with the required level of authentication is beyond the scope of this document.

4 Other information

4.1 General

Decentralised draw systems are primarily based on the following concepts and technologies:

- Secure Hash Algorithms (SHA) (also referred to as a *fingerprint* – where the task of reverse engineering a hash result to reveal the original hashed data is extremely difficult).
- When XOR combining bits of variable entropy from independent (physical) sources, the rule of strongest applies i.e. the resulting overall entropy only increases.
- Security in numbers (aka the multiplicity of draw participants).
- To change the outcome of a witnessed event, all witnesses (aka draw participants) must be compromised.

An important facet of decentralised draw systems is how each draw participant generates sufficient bits of entropy for the draw. Before disclosing their entropy file to other draw participants, the participants first exchange only their draw entropies' fingerprint. This prevents draw result manipulation by any draw participants. The security here is as good as the specific Secure Hash Algorithm utilised during the draw process. A SHA-256 or stronger should be used.

Once results are shared between all draw participants, the results are all combined via a simple bitwise XOR operation.

With XOR combination of binary entropy from independent sources, the rule of strongest applies i.e. the resulting entropy level from XOR combining independent sources of entropy bits will be at least as good as the strongest source of entropy bits in the group. For example, in a given draw, if all the draw participants' entropy files were compromised or weak bar one draw participant, then the entropy level / integrity of the final XOR combined draw entropy file remains untarnished given each draw participant provided sufficient bits of entropy for the draw.

In a decentralised draw system, for a draw to be compromised, **all** draw participants must be compromised. Given a sufficient number of draw participants this would be extremely difficult to achieve, and the difficulty only increases the more draw participants there are. This is the primary source of security in a decentralised draw system – the number of draw participants. Security is also boosted by the level of independence between draw participants and the respective hardware and software used in their draw devices.

Refer to the previous section for the detailed draw process encompassing these concepts.

4.2 Draw participants

A DDS is comprised of at least two draw participants. DDS draw participants may be people, computers or organisations. Each draw participant is effectively a witness to each draw as well as directly participating and contributing entropy from their own draw device each draw. Individual entropy contributed from each draw participant is required to be sufficient to conduct the draw alone and is not designed or intended to increase the overall entropy of the overall draw. The extra entropy in a DDS is directly related to security, not randomness.

The more draw participants there are, the more secure the draw outcome will be, as **all** draw participants must be compromised for a draw not to be 100% secure.

There must be sufficient draw participants commensurate to overall draw risk.

Who should be a draw participant for a given draw? Recommend:

- The primary operator associated with the draw.
- Draw related auditors, inspectors or their respective organisations.
- Any person or organisation with a vested interest in the integrity of the draw where possible. In gaming applications the players should be draw participants where this is applicable and convenient since they inherently have a vested interest in the draw outcome.
- A DP who has no vested interest in the draw outcome can be useful in relation to reducing the risks of a denial of service attack where applicable. Refer section 4.5.

Almost any potential draw participant that can deliver the desired level of availability and reliability is a good draw participant. The more there are the better it is for overall security. The more independent each draw participant is with respect to other draw participants, the more they also contribute to overall draw security.

Who should not be a draw participant?

- Anyone that may have an interest in seeing a draw **not** complete for any reason at any time.
- Draw participants with reliability or availability issues that could affect a draw from taking place at the desired time.

Draw participants who are separated from each other (in any respect), contribute more security than draw participants that are less separated or are sharing resources.

How many draw participants is enough?

The number of draw participants should be commensurate to the required level of security and overall risk of the draw. A risk assessment of individual draw participants may be considered, but when a draw participant is an organisation, then without any further probity or security assessment, one organisation should be counted at best as the equivalent of one human witness to the draw.

Other considerations:

Outsider observers or auditors of a DDS need only establish trust in a single draw participant in order to be able to trust the overall DDS draw outcomes.

While draw participants should openly share best practises concerning security and draw device design, it is not unreasonable to allow draw participants to keep the technical specifications and specific security arrangements pertaining to their draw device a secret. This makes it harder for any would be attacker to successfully attack all draw participants (which is required in order to defeat a DDS).

Each draw participant need only be primarily concerned with their own security. Any overall security concerns in a DDS should be addressed by adding more draw participants.

4.3 Caveats

Manual (human driven) decentralised draw systems still require a device to act as an RNG and perform the hashing and XOR operations required as a part of the draw process. Any current generation Windows / Linux based desktop computer using off-the-shelf utilities should suffice.

The RNG utilised by each draw device / participant must predominantly involve a **hardware based RNG**[v] (or hardware based entropy) in random number generation. This is because that when combining random numbers via XOR, it is important that the random numbers are independent. Hardware based RNGs will generally ensure this independence. Combining the random numbers of the output of two or more PRNGs is not recommended as unwanted and difficult to detect correlations may occur. However, combining the outputs of two hardware RNG's is fine (demonstrably a hardware RNG bit-bias software whitening technique)[vi]. Further research here may prove wise to restrict draw participants from utilising some or all forms of software based whitening in their RNGs because of the XOR operations later applied as a part of the DDS draw process.

Each draw device in a decentralised draw system must be fully capable of conducting the draw by itself i.e. able to produce the required amount of entropy bits for any given draw. The intent regarding the participation of multiple draw devices in a decentralised draw system is to contribute to draw security. It is not for the purpose of boosting the amount of entropy for a given draw.

The SHA fingerprint and RNG algorithms should be upgraded over time as needed. Only algorithms with a proven industry and academic track record should be used. Currently **a SHA-256 or better algorithm must be used**.

Where human draw participants are used, ideally they should understand,

- the basic concept of the XOR operator;
- what a SHA fingerprint represents (and how to make them);
- what a digital signature represents; and
- how to use the various software tools required to participate in a draw.

It is possible to develop a user friendly GUI application to perform complicated tasks for less skilled draw participants; however, this will introduce a need for a custom application to be developed and authenticated. Less skilled human draw participants will still require a way of either establishing trust in the application tools they use, or a method to indelibly prove they used the application.

The DDS *draw parser* (which can be either a script, a procedure or a set of rules) which produces the final draw file must be agreed upon by all draw participants before draws commence. As DDS draw parsers are typically shared as source code, and perform a deterministic process, mean they are fairly easy to test. All the other draw devices, tools, RNGs, hardware, software and security arrangements can vary between the organisations. Variations here enhances DDS security because it makes it harder for an attacker to successfully attack all draw participant's systems.

[v] Refer: https://en.wikipedia.org/wiki/Hardware_random_number_generator

[vi] Refer: https://en.wikipedia.org/wiki/Hardware_random_number_generator#Software_whitening

The separation (in all respects) of draw participants and their draw devices in a decentralised draw systems can contribute to overall security of the DDS.

In a decentralised draw system, provided one draw participant is not compromised then the integrity of the draw is 100% regardless of the other draw participants.

It is generally important for verification purposes that draw participants each ensure that the final draw results have actually been used / published in the overriding draw system or game. This specific task is outside the scope of this document.

It is recommend that all draw participants publish their results individually in order to allow anyone to reconcile the results as they see fit.

The use of different operating environments (operating systems, API's, applications, security etc.) is encouraged between draw participants. This helps prevent a single vulnerability from being exploited across all draw participant's systems.

4.4 Benefits of decentralised draw systems

Decentralised draw systems have the following benefits:

- Decentralised draw systems can implement an **infinitely scalable level of integrity and security in order to meet any given level of operating risk.** The principle of strength-in-numbers applies with decentralised draw systems. Security in a DDS is increased by adding more draw participants. Each draw participant is akin to a witness to the draw as well as participating in it.

- As the security of decentralised draw systems comes from having multiple draw participants working in the manner described by this document, the security of individual draw participants and draw devices in a decentralised draw system can be decreased in certain areas to **reduce costs**.

  Each draw participant need only be primarily concerned about the security of their own draw device. Given a sufficient number of draw participants in a DDS, draw devices can simply be incorporated into the hosts they serve at the host system's given level of security and don't have to be tamper sealed.

- DDS **costs** (setup, security, operation) can be significantly less for any given level of risk/security than compared with traditional centralised draw device based solution for the same level of security.

- Decentralised draw systems are **highly auditable** and are also **self-auditing** among the draw participants. The DDS draw process itself is also highly transparent.

  Auditing of draw results in a decentralised draw system is not necessarily required as the draw process itself is representative of an audit given the multiple draw participants. The main audit related requirement in a DDS is to simply ensure that the results drawn are actually the results used (Ideally each draw participant's results should be published, allowing anyone to reconcile results with other participant's results and with the final result).

- Decentralised draw systems can be fully **automated**.

- Draw participants can be **located remotely** from each other. This benefit has many flow-on applications. Refer to section on applications for more information and examples.

- Draw participants who have never met before can immediately create **draw outcomes all can implicitly trust**. It is unnecessary for draw participants to perform a probity checks on each other.

- **Rule of strongest applies**. In a computer system, a general tenet is that security is only as strong as the weakest link. However, in a DDS the opposite applies. The security and integrity of a DDS draw remains at 100% as long as at least one draw participant / device is not compromised. Attacks on decentralised draw systems cannot succeed unless every single draw participant is successfully attacked.

- A DDS has **built in overall resilience** to attacks in proportion to the number of draw participants and level of diversity and separation between them.

Each participating draw device can be based on different hardware, software and operating systems to ensure that no common single weakness can be exploited in order to defeat all participating draw devices.

- Decentralised draw systems can also be arranged to have **built-in fault tolerance** and redundancy against communications and other failures. A DDS could be setup to have a set of agreed mandatory participants and a range of optional draw participants. Typically any arrangement in this regard would be agreed prior by all draw participants when the draw system was being initially setup. Each draw participant would also have a number of draw devices on standby for redundancy purposes. Note, changes to the arrangement of draw participants or their draw device / entropy cannot occur once the entropy sharing stage has begun.

- **DDS are suitable for emergent / ad-hoc draws**. It is possible to setup and conduct high risk draws using only human draw participants[vii] utilising off the shelf personal computers and software applications.

- **A worldwide market for draw participants.** Cooperating[viii] governments, organisations, individuals or their servers, anywhere in the world could offer draw participant services, allowing the easy creation of worldwide decentralised draw systems that could not be compromised in any feasible scenario.

- **Decentralisation.** DDS draws (including win verification and audits) can be conducted remotely.

[vii] Human draw participants do require some computer skills and competencies to participate in a manual draw using the decentralised draw system. However custom software could remove this requirement.
[viii] The willingness to participate and to publish results is the only cooperation required among draw participants.

4.5 Issues / risks

This section discusses possible issues and risks with decentralised draw systems.

- **Denial of service attacks and related risks**

  With typically remotely located draw participants, decentralised draw systems can be vulnerable to network based denial of service attacks. While draw participants should be transparent and publish their draw results, in order to avoid denial of service attacks, draw participants can keep the networking details used to conduct a draw a secret among themselves.

  A higher risk is that any individual draw participant can prevent a draw from occurring by simply refusing to participate at any stage during the draw. This means that there is an obvious caveat that draw participants must be inherently willing to participate, otherwise it may be necessary to create some additional redundancy and rules for non-responsive draw participants.

  Issues can also arise during the process of exchanging *entropy file*s in the case where a draw participant withholds their file until they have all other participant's entropy files. This draw participant can calculate the final outcome first and if they don't like the result, then refuse to disclose or destroy their entropy file and deny draw completion.

  One way to reduce this risk is to require draw participants to use a pass-to-the-left or similar technique for the sharing of *entropy file*s. In this way, each entropy file is shared with at least one or more other draw participants before the final outcome can be ascertained. Alternatively, it may be prudent to introduce more draw participants that have no vested interest in the draw result and agree (before the draw commences) on a designated order in which entropy is shared.

- **Issues in relation to manual / human conducted draws:**

  Speed

  In human based decentralised draw systems, the more draw participants there are, the longer the draw takes due to the number of fingerprints that have to be exchanged and checked during the draw process e.g. each SHA-256 fingerprint is 64 characters long, which takes roughly 40 seconds to read aloud. An electronic means of communication and text manipulation tools are therefore highly recommended for manual draws. A diagrammatic alphabet could also be used to help speed up the process.

  Required competencies of human draw participants

  The DDS manual draw process may be too difficult for non-technical draw participants without additional training. An alternative to training or the use of IT professionals as draw participants, could be to produce a user friendly application to enable the use of non-technical draw participants.

- **Draw poker and similar applications**

  Most common variants of draw poker and similar games and applications **do not work** with decentralised draw systems. This issue is specific to games and applications that operate along the lines of:

  - pick X with no replacement,
  - keep X **secret**, and
  - pick Y more

  In some cases, if you can remove any one or two of the above listed game facets, then the game / application can work as a DDS. For example, in draw poker, if an infinite deck is used (i.e. remove the ‘no replacement’ facet), then players can play remotely using a DDS where each player is a draw participant.

- **Use of Exclusive-OR (XOR)**

  The use of XOR for combining entropy from each draw participant has the advantage in that the final combined draw entropy is not dependent on the order of combination. However the use of XOR in the DDS process means that each draw participant must utilise a predominantly hardware based RNG, as it is considered bad practise to XOR the output of multiple PRNGs.

  As a result of the use of XOR, it may also be prudent to ensure that entropy is only whitened[ix] once per draw in a DDS depending on the whitening algorithm used. Making whitening a part of the DDS draw parser would alleviate any risk here. Some whitening algorithms do not appear not be at risk. Further research may be required.

  A variation on a DDS where the draw participants supply only the entropy to seed a strong PRNG located in the DDS draw parser may be worth considering for some applications. This would be another way to alleviate any possible concerns related to the use of XOR in the DDS process.

4.6 Draws containing secret results

This subsection demonstrates how to implement DDS applications where there is a need to pick X random results (**with replacement**) and keep those results a **secret** from one or more draw participants (DPs) until the end of each play. Blackjack (using an infinite deck) is an example of a game that fits under this arrangement.

In order to facilitate an application such as this, the DDS draw procedure requires a small modification. The explanation in the next paragraph, assumes the reader is familiar with the DDS procedure in section 3.

Each draw participant wishing to generate results but keep them secret generates two entropy files and hashes. One entropy file and corresponding hash is made and shared as per the DDS procedure in Section 3. For the other entropy file, it is still

[ix] Refer: https://en.wikipedia.org/wiki/Hardware_random_number_generator#Software_whitening

combined with the shared entropy from other DPs and its hash shared, but the actual entropy file is not shared until the play is finished. This second entropy file is used by the DP to create their secret results via an agreed shared draw parser. The sharing of its hash however, prevents the DP from altering their secret results.

Nothing should be finalised or paid until the end of the play and all secret entropy is finally exchanged thus allowing all DP’s to verify no other DP has cheated.

This also may require up to two draw parsers - one for creating shared results (if there are any) and one for creating a DP’s secret results.

Finally there needs to be a standing rule for dealing with DPs who fail to disclose hash matching secret entropy at the end of a play, such as disqualification, or forfeit of bet, etc.

5 Conclusions

- Decentralised draw systems can be setup to meet the security needs of any draw / random number generation system.

- Decentralised draw systems have scalable security which increases in proportion to the number of draw participants. Extra security (by the addition of more draw participants) is linear in cost on average and is uncapped.

  The more separated the draw participants are from each other, in design, as well as business and relationship wise, the more security each draw participants adds to the DDS.

- The overall level of security attained in a DDS is easier to evaluate than traditional centralised draw systems since it can be initially estimated simply based on the number of draw participants. Depending on risk and operating environment a light weight probity check of each may be considered. If the outcome of a DDS risk assessment is borderline, then arguably there are simply not enough (independent and diverse) draw participants.

- Decentralised draw systems can implement automated draw systems with high levels of security more cheaply and with more security than other draw systems trying to achieve the same level of security by traditional means.

- Decentralised draw systems are well suited to both manual / human based and automated draw arrangements.

- Decentralised draw systems are easily understood.

- Each draw participant in a DDS only needs to maintain trust in their own draw device and security in order to satisfy themselves of overall draw outcome integrity. There is no need for one draw participant or third party to verify the draw device of another draw participant. In a DDS, provided there is one good draw participant, the draw result is secure. Security in a DDS comes from the physical, logical, organisational and business separation of multiple individual draw participants and the diversity between their respective draw device hardware and software (RNG). The less each draw participant knows about any other draw participant's draw devices the better (or any single entity for that matter). If further security is believed necessary for any given application, it is simply a matter of adding more draw participants.

- Draw participants are free to keep their entropy generation methodology a secret and vary it (even if only marginally) in order to avoid potential vulnerabilities being shared or exploited across multiple draw participants.

- The high levels of security achievable by decentralised draw systems can potentially alleviate the need for:

  - multi-party commissioning of draw devices. (Typically a long, complex and resource expensive process).

  - Physically tamper sealing RNGs / draw devices and perform subsequent regular seal inspections.

    - Source code reviews, evaluations and typically complex build verifications of RNGs / draw devices.

    - Making RNGs a discrete physical device with respect to the gaming host it serves.

- The creation of an open standard network protocol for use among draw participants in automated decentralised draw systems is highly desirable in the progression of the DDS concept.

- DDS can be used as a manually / human driven ad-hoc disaster recovery draw system (Suitable for low frequency use only).

- A DDS eliminates the significant and costly-to-eliminate risks associated with insiders and people with privileged access and knowledge in secure draw systems.

6 Applications

Decentralised draw systems can be used in almost[x] any application which requires random numbers.

Possible applications include, but are not limited to wagering related applications such as lottery systems, casino games and poker machines. General computer gaming and board gaming can also benefit. A DDS allows players located remotely to play games that require an RNG without having to establish the repute of the other players. Without a DDS, remotely located players must either trust each other, or a third party, with respect to the random outcomes needed to play the game.

A game-like application where the players (or the equivalent) are also the draw participants is an excellent arrangement and this should be done whenever possible.

Decentralised draw systems are ideal for any application where the draw participants are unknown each other, or when a third party dealer or draw device is unavailable or not desirable.

6.1 Lottery systems and keno

Decentralised draw systems are perfectly suited as lottery system draw devices and all similar applications. Lotteries, Keno and similar applications were the applications in mind when the DDS concept was initially drafted.

6.2 On-line casinos

[x] Any application where the draw results do not have to be kept secret for any period. See the “draw poker” section in this chapter

In relation to on-line casinos and gambling, by using a DDS where the player forms a DDS with the on-line casino, casino games could now be played on-line without the player having to solely trust the on-line casino to determine game outcomes.

6.3 Mission critical applications

Decentralised draw systems can be used for determining the outcome of mission critical draws that require the highest possible level of security and transparency.

A decentralised draw system where each party with a vested interest in the outcome is also a draw participant is the best arrangement for draws pertaining to mission critical applications.

For example, if two or more countries are in dispute and want to randomly decide on an event or outcome, then a decentralised draw system with each country as a draw participant, ensures the draw can be conducted in a transparent manner (and remotely), without any possibility of dispute over the outcome by the participating countries.

Each draw participant need only trust their own draw device in order to satisfy themselves of overall draw outcome integrity.

6.4 Poker machines

In this section poker machines (aka gaming machines[xi]) refer to machines running spinning reel games and all other games but excluding games that need to keep secrets such as draw poker and variants.

Poker machines running most games (with respect to Section 4.6) could potentially make use of decentralised draw systems but not without significant changes to the current operating environment in which poker machines operate. For this reason this section is purely hypothetical. Summary:

- A fully automated poker machine DDS is potentially viable (see below), including self-verification of all wins among all draw participants.
- A player account based gaming system should be in use as it is key to ensuring that prizes won are correctly paid by the poker machine. It would be a requirement that the player account based gaming system or operator is a separate entity from the poker machine manufacturer.
- The notable impediment to a fully automated poker machine DDS is that certain game data is kept a trade secret for copy protection reasons.
- A poker machine DDS is also one of the more complex of all DDS applications to make work. Its feasibility depends on the industry's value of the potential advantages (see below).

In a poker machine DDS, it should be an arrangement where at least some of the draw participants are unrelated to the manufacturer of a poker machine. For example, potentially good arrangements of draw participants concerning poker machines occur where each

[xi] AU specific term.

poker machine forms a DDS with draw participants consisting of itself and other local machine of the manufacturer, and other stakeholders; possibly:

- other brands of machines at the local gaming venue;
- the gaming venue operator;
- the player account based gaming system. (They are a key / mandatory draw participant as only they can easily verify the correct payment of the wins to players.)
- other cooperating gaming venue operators;
- a government regulator or other government licensed authority.

As mentioned in the conclusion section, an open standard DDS protocol is highly desirable in the progression of this concept.

Specific additional advantages and caveats pertaining to a poker machine DDS

A DDS in poker machines implemented with a sufficient level of security could potentially allow other resource heavy forms of security to be either relaxed or removed, incurring significant cost savings. For example, the requirement for poker machine source code evaluation and build verification; the requirement for tamper sealing of poker machines and subsequent inspections of those tamper seals and poker machine hardware.

In relation to the *draw parser* in poker machine based DDS (refer section 3.1):

The draw parser could also know how to scale and map the random numbers to the game's symbols. Or better; the parser could output the overall prize result value making for easy prize verification and auditing. Unfortunately however symbol layouts are generally a trade secret which would prevent this.

One possible way to allow game manufacturers to keep certain game data a trade secret and still allow any party to be a draw participant, is to have a simple draw parser shared among the draw participants, one that only outputs the final draw entropy data, and have a restricted full blown draw parser (that decodes all the way to the prize value) provided just to draw participants the game manufacturer trusts; such as gaming regulators and other specifically authorised organisations. In this arrangement only draw participants with the full blown parser can verify a game result down to the prize amount, but all draw participants can still verify that integrity of the raw draw entropy data.

One huge advantage of a draw parser that outputs the final prize value, is that poker machine evaluation and approvals could potentially be reduced to just the evaluation of the draw parser. The regulator simply approves the game maths and game draw parsers. The efficiency improvement here is that the game manufacturers would not need to provide the regulator (or delegated testing authority) with a test poker machine and associated hardware, software, development environment, compiler, emulator etc. The main reason that this is possible is because a poker machine DDS extrudes game outcome determination from the poker machine and distributes it among the draw participants, thus the need to have a fully secured poker machine is substantially reduced. Draw participants only need to trust the DDS draw parser to fully trust the outcomes of a poker machine DDS.

6.5 Web browsers

One possible embodiment of a standard DDS protocol would be to incorporate it into **web browsers** and make every web browser a potential draw participant on behalf of its user. This is useful because most people already trust their browsers implicitly, for example internet banking. A standard DDS protocol could be used by an **online gaming operator** to setup a DDS between the gaming host and the player's browser. This arrangement could demonstrably prove to a player (who trusts their browser) that the game results they are seeing are random and not biased in any way.

Decentralised draw systems can be used in the above arrangement to help make the online gaming industry more secure and transparent.

However this DDS-in-a-browser concept does have a number of specific challenges that need resolving before it could be realised. For example:

- How does the browser establish trust in what would need to be the *draw parser*? (Refer section 3.1.)
    - Many games are standard and well known; there could be a standard set of draw parsers built into the browser.
    - For a range of games (like roulette and keno for example), a raw scaled number result display may be sufficient (provided the range/s is also displayed and a session/draw serial number[xii]).

- How does the browser display work with the on-line game server? The browser could for example:
    - exclusively and indelibly understand and display the results to the player using browser controlled code, or,
    - Interpret and verify the results it knows the player is seeing and raise an alarm upon any discrepancy (something similar to the www certificate ok padlock display currently used by browsers) or,
    - Co-display of the results to the player along with the operator's results for on demand verification by the player.

  Alternatively, the DDS browser embodiment might be easier to achieve if instead of the DDS draw parser stopping at draw results, it works out the win amount (for this it would need to also know the bet amount and pay tables). Potentially this approach is a lot easier to verify as it is just a number (a credit meter or wallet value), rather than any given game's result display which is highly variable. Another advantage of this approach is that the parser and therefore the browser would know the game's percentage return as well as the game's standard deviation (volatility). This additional information if displayed could help players make more responsible on-line gaming choices.

A rough example of a DDS enabled web browser URL display may look like:

```
<padlock symbol>Big Bucks Pty Ltd. | Roulette 98.6% $1234.56 | https:\\www.bigbucksgaming.com
```

[xii] The browser would need to detect or make evident a gaming server that tries to spam draw results until it gets a result in its favour.

Double clicking on the green section may bring up a help screen which may include information such as:

The browser is participating in game draws at this site to help ensure your game is random. You are playing Roulette, a game this browser understands. For every $1 bet, your average win is 98.6 cents. The amount at $ value should match your current credit meter / game wallet value at the end of each play. If not you should stop playing immediately and contact the site for assistance.

6.6 Servers in general

Not an applicable application for a DDS.

Servers around the world have a big demand for secure random numbers. However, as servers typically need to keep the entropy they produce a **secret,** this makes the use of a DDS of no use to the point where by a DDS could actually compromise the server's RNG security.

A DDS is of little to no use in applications where the final entropy / outcome must be kept a secret.

6.7 Ad-hoc / disaster recovery draw systems

Decentralised Draw Systems are perfect for Ad-hoc / disaster recovery / once-time or low frequency draws.

Appendix A – Notes for manual draws

This Appendix contains notes regarding possible tools which might be considered for use with an ad-hoc or manual i.e. human driven decentralised draw system. For example, for use as a disaster recovery draw system in applications where draws are infrequent enough to allow manual draws to take place. Humans used in a manual draw process will need to be proficient in all the technology listed below.

Hardware requirements:

- A computer in which the draw participant can establish trust commensurate to the risk associated with the draw is required. Its tasks are implementing the RNG, hashing and communication. Differences in the computers, OS and software used in a DDS is beneficial because it contributes to the security by helping to eliminate common vulnerabilities.

- The computers also need to be networked.

- A source of hardware based entropy. See RNG requirements below.

Software requirements:

- Text based group chat software (For the easy exchange of entropy file fingerprints).

  There are many freely available options here.

- Small file transfer/sharing software. (We are talking ~4-10kb for the exchange of draw *entropy file*s)

  Alternatively draw participants could solely use the text based chat program to share the entropy files when binary to text encoding software is also used.

- A random number generator.

  The RNG must incorporate some form of hardware/physical based entropy source for use with decentralised draw systems to ensure independence between all the RNGs in the DDS.

  One approach is to use a strong hardware based entropy combined with a good software based PRNG (such as the Mersenne Twister based software RNG, or an encryption algorithm). Note: since 2013, Intel processors contain a built in hardware RNG[xiii] and since 2015 for AMD processors.

  The Linux operating system can provide secure entropy without any custom application (/dev/hw_random[xiv]) and there are also plenty of freely available sources of entropy from the internet. These could be combined to make a better random file[xv]

[xiii] https://en.wikipedia.org/wiki/RdRand

[xiv] *Note historically hw_random can be very slow and is often supported with entropy from external sources*

[xv] FYI copy to /dev/random and Linux will incorporate the data into the OS entropy pool.

- Hashing / fingerprint utility

  There many freely available options, e.g. “Hashcalc”, “hashmyfiles” or “openssl” to name a few.

- File merge via XOR utility

  This is not a readily available application. However, a tool to perform this operation is simple to create and test, just a few lines of code in any high level scripting language.

  A scripting language interpreter which can parse a random number file into human readable formatted final draw results. The file should be in a format which is readable by both a human and a computer (e.g. ini, csv, json, txt, xml etc). It is recommended that the draw parser is coded in a memory managed high level language (e.g. java, Python, Lua, etc.).

Appendix B – Generating random numbers

This appendix is not required reading.

This appendix consists of some notes in relation to some ad-hoc techniques for manually generating files of strong random numbers using the internet and a few freely available software tools – i.e. without having to implement an RNG application or draw device or have any major technical knowledge or training.

This appendix can be regarded as a brainstorm into the feasibility of training a non-technical person to gather entropy bits using just a PC or laptop they trust.

Possible readily available entropy sources:

- Use an entropy provider like www.random.org.
- Use Linux (hw_random).
- Use application/OS with a computer which supports the Intel RDRAND instruction.
- Digital cameras (an unlimited source of high entropy pictures.)
    - Low quality digital cameras are also good (ones that provide grainy images in low light environments).
- Microphones.
- A wide range of day-to-day volatile internet content can be used.
- Human input devices such as mice and keyboards.

Mix up the above options by injecting or combining them with any number of other ‘whitened’ readily available sources of entropy readily available from the internet.

Never trust a single source of entropy.

The possible sources of entropy available from the internet is virtually limitless:

- Pictures
- Music
- Articles
- News and weather reports and information (audio, video, text)
- Online Webcam pictures or videos.
- Snippets of live streams.
- Thermal noise from web-cameras viewing dark areas
- Published results from lotteries.

Pictures and videos with a lot of “noise” like images are better, e.g. videos of tree leaves, water in motion and fire are arguably media with high entropy e.g. a video of a power-point presentation or computer animation would be a less effective source of entropy.

Most sources of hardware and environmental entropy should be “whitened” before use. When combining entropy (as in a DDS) it may be prudent to ensure that entropy is only whitened once depending on the whitening algorithm used.

In a DDS, participants have to be careful not to use the same source of entropy and should try to use at least one source of entropy that they know is unique.

One of the challenges with regard to gathering entropy from data on the internet is estimating how many bits of entropy the data actually represents. The extent to which the data compresses can be an indicator.

Appendix C – Miscellaneous references and notes

RNG ‘whitening’:

http://en.wikipedia.org/wiki/True_random_number_generator#Dealing_with_bias

Entropy in computing: http://en.wikipedia.org/wiki/Entropy_%28computing%29

Salt (cryptography): https://en.wikipedia.org/wiki/Salt_(cryptography)

XOR merge programs:

> http://trillian.mit.edu/~jc/src/misc/xor.b
> NB if files are of different lengths, then depending which is shorter then code may abort or reopen the short file from the start and continue
>
> http://monolith.sourceforge.net/
>
> Related:
>
> http://stackoverflow.com/questions/6889067/how-to-perform-bitwise-operations-on-files-in-linux

Free random bits for draws or RNG seeding from: http://www.random.org

Random numbers and random number in Linux:

> https://calomel.org/entropy_random_number_generators.html

Linux: Monitor the amount of available entropy:[xvi]

```
watch -n 1 cat /proc/sys/kernel/random/entropy_avail
```

A Pseudorandom Number Sequence Test Program: http://www.fourmilab.ch/random/

RDRAND CPU instruction:

> http://en.wikipedia.org/wiki/RdRand

Example of how to collect some random numbers (1024) into a file in Linux:

```
dd if=/dev/random of=outfile bs=1 count=1024
```

> “Warning: this can take a very long time even for small files when the device is based on the a hardware based RNG and the machine doesn’t many sources to gather additional entropy.”

[xvi] NB Warning; every time you run a program in Linux, the system’s available entropy takes a noticeable hit.

“If Linux sees a hardware RNG, there may be a device: (/dev/hw_random) but don’t use it, Linux will auto incorporate into /dev/random”
“NB you can also write “random” junk to /dev/random to help create entropy to speed things up. The internet is a huge source of entropy. Linux will hash data copied into the device into the entropy pool so it doesn’t need to be whitened prior. Also simply using the computer, e.g. browsing the internet, tasking the hard disk, using the mouse and keyboard all help modern computer operating systems generate entropy.”

Ref: https://www.kernel.org/doc/Documentation/hw_random.txt

Revision history

Version	Changes	Who	Publication date
NA	Invented March 2013	Robert Larkin	NA
0.1 draft	Initial draft – commenced 2 Oct 2013	Robert Larkin	NA
1.0	First release / published Edocs ref: # **1681220**	Robert Larkin	15 March 2018